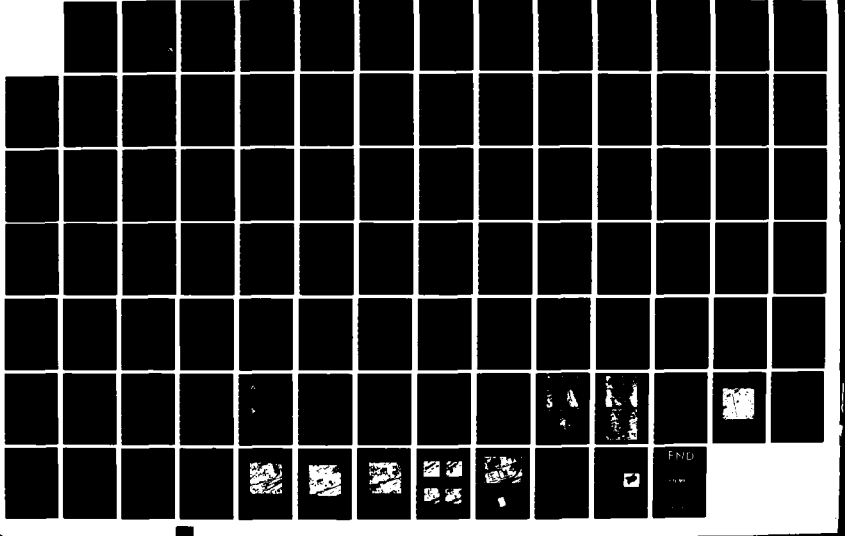
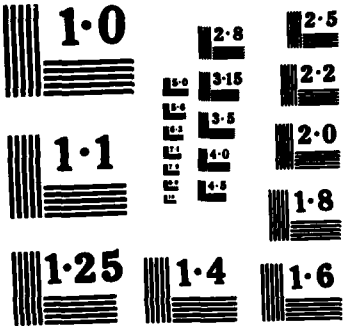
AD-A157 913 CORRECTING A DIGITAL MAP DATA BASE WITH SCENE ANALYSIS CONCEPTS AND METHO..(U) FORSCHUNGSZENTRUM GRAZ (AUSTRIA) INST FUER DIGITALE BILDVERAR.. H RANZINGER 1/3

UNCLASSIFIED MAY 85 R/D-4161-R-EN DAJA45-83-C-0022 F/G 8/2 NL





RADC-TR-84-161-REEN

CORRECTING A DIGITAL MAP DATA BASE WITH SCENE ANALYSIS CONCEPTS AND METHODS

HUBERT RANZINGER
Graz Research Center

85 7 08 111

The research reported in this document has been made possible through the support and sponsorship of the U.S. Government through its European Research Office. This report is intended for internal management use of the contractor and the U.S. Government.

CORRECTING A DIGITAL MAP DATA BASE WITH SCENE ANALYSIS CONCEPTS AND METHODS

HUBERT RANZINGER
Graz Research Center

Principal Investigator
Hubert Ranzinger
Graz Research Center

Contract Number DAJA-45-83-C-0022

Contractor
Graz Research Center Joanneum
Institute for Image Processing
and Computer Graphics

Interim Technical Report
May 1985

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) CORRECTING A DIGITAL MAP DATA BASE WITH SCENE ANALYSIS CONCEPTS AND METHODS		5. TYPE OF REPORT & PERIOD COVERED Interim Technical Report 1.1.1983 - 31.12.1984 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) HUBERT RANZINGER		8. CONTRACT OR GRANT NUMBER(s) DAJA 45-83-C-0022
9. PERFORMING ORGANIZATION NAME AND ADDRESS Institute for Image Processing and Computer Graphics, Graz Research Center Wastiangasse 6, A-8010 GRAZ, Austria		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS U.S. Army Research, Development and Standardization Group, U.K., 223 Old Marylebone Road, London NW1 5TH, England		12. REPORT DATE MAY 1985 13. NUMBER OF PAGES 91
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) 15a. DECLASSIFICATION DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Available for Public Release-Distribution Unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Feature Recognition, Map Updating, Expert System

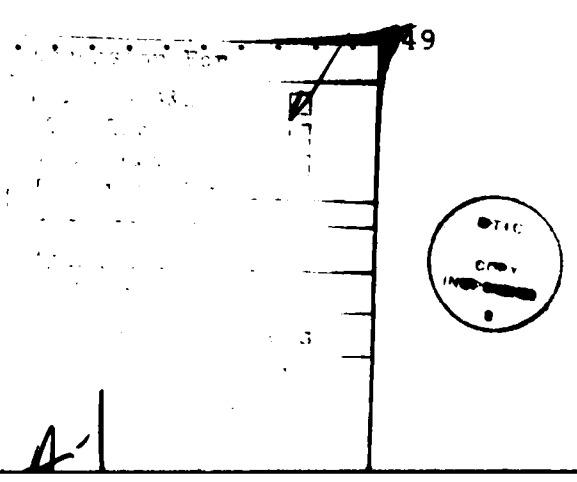
20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The problem of updating maps has always been of considerable interest for cartographers. With the advent of digital cartography and the supply of aerial imagery in digital form in the near future, the question arises in which way information to update maps may automatically be derived from digital imagery by means of scene analysis. This paper proposes an approach and show how a solid set of methods, modularly designed and based on an image processing language, may serve as a toolkit for experimentation. Examples are given by applying the methods to a multitemporal sequence of digitized metric aerial photos.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

TABLE OF CONTENTS



Abstract

The problem of updating maps has always been of considerable interest for cartographers. With the advent of digital cartography and the supply of aerial imagery in digital form in the near future, the question arises in which way information to update maps may automatically be derived from digital imagery by means of scene analysis. This paper proposes an approach and show how a solid set of methods, modularly designed and based on an image processing language, may serve as a toolkit for experimentation. Examples are given by applying the methods to a multitemporal sequence of digitized metric aerial photos.

1.0 Introduction

The field of applications of computer sciences is rapidly increasing. At the beginning, computers were used in the original sense of the word, namely to calculate numerical problems. Soon the question arose whether these machines were capable of performing "intelligent" tasks which go far beyond purely mechanical procedures. This lead to the emergence of a new branch in science: Artificial Intelligence (AI).

The goal of artificial intelligence is to propose and develop methods which make use of a computer's capabilities to process information similar to biological organisms. Here, one of the information sources is visual perception, with which computer vision in concerned. Computer vision is, after Ballard and Brown (1982), the "construction of explicit, meaningful descriptions of physical objects from images".

Modern imaging systems acquire image-like representations of the world already in digital form. Much effort has been put into information extraction from these data alone and has yielded a solid basis of digital image processing methods. However, human perception of the world involves knowledge, which is mostly acquired by learning and subsequent deduction. The issue in artificial intelligence is therefore to make knowledge in some form also accessible to automata.

1.1 Expert Systems and Knowledge

The incorporation of knowledge in a program leads to so-called expert systems. Nau (1983) gives an overview of the concepts involved. The model for problem-solving is stated explicitly in a knowledge-base. This may be termed as propositional or descriptive representation as opposed to procedural knowledge where the program code itself contains the strategies to be taken.

McCalla and Cercone (1983) name the following approaches to knowledge representation

- semantic networks
- first-order-logic
- frames

The knowledge-base is manipulated by a separate control strategy. Of course, on a high level, the control structure itself, as it is a program, incorporates again procedural knowledge, namely how to handle the knowledge-base, and thus limits the set of actions which can be made. Thus, today's expert systems are constructed with respect to particular applications, at present predominantly in medical consulting and in natural language understanding. The general layout of an expert system is depicted in figure 1. This report will treat in-depth the ways in which the method-base is established; the problems of building a comprehensive

knowledge-base will require more insights as to the nature of expertise involved in photo-interpretation.

1.2 Specification of the Problem of this Study

In this study, we are concerned with one special aspect of computer vision: How can a digital map serve in automatic image interpretation, and, on the other hand, how can interpretation results be used to change or update the map? In a wider sense, "map" may mean any graphic representation of a scene that is imaged. Here, in particular, we deal with maps in the cartographic sense, and with images from airborne photographic systems. One of the obvious applications is the correction or densification of a map data base using time series of aerial surveying imagery.

The aim of this study therefore is the design of a strategy to evaluate the usefulness of image-map correspondence to aid the interpretation of digital aerial photography and to develop methods which form a testbed for further studies. This is the first step to be taken towards a photo-interpretation expert system, which we shall henceforth name PHIX.

Aerial photography is one source for the update of cartography. It is acquired on a regular basis, however, the updating for many map series is, as a rule, several years. Support in the interpretation of the imagery can be given by focussing on changes rather than on invariant

information. Thus the attention of the human interpreter can be directed to relevant locations in an image and, in a next step, supplying hypotheses about the nature of the inconsistencies between map and image. He then can interactively work on the data indicated and enter his interpretation in a suitable form.

1.3 A Review of Literature

Several efforts have been described in the literature to use a map data base to analyse aerial images.

At the Stanford Research Institute, Barrow et al. (1977), Tenenbaum et al. (1978) or Fischler et al. (1979) used map data to guide feature detection. Roads or coastlines were identified by predicting their locations and thus restricting the search in the image matrix to small areas where elaborate pattern recognition methods could be applied.

Lantz et al. (1978) describe an approach taken at the University of Rochester. A semantic network is used to represent declarative and relational knowledge of the image contents. The nodes in the network describe which procedures are to be performed during interpretation.

At Carnegie-Mellon University, McKeown (1982) and McKeown and Denlinger (1982) report on a semi-automatic image understanding system which relies on a pictorial database, a map data base and a rule base, where the rules have general

knowledge about objects of the real world rather than present particular facts about specific objects. A first application - the segmentation of airport scenes - shows the feasibility of this approach, though the very general concept pays - at the current state of available computing power - a heavy computing time penalty.

Havens and Mackworth (1983) from the University of British Columbia describe the Mapsee2-system which uses schema models in a network. Each model represents a class of objects, providing a description of the generic properties of every member of the class and specifying possible relationships of the class with other schemata in the network. With this knowledge, a structural description of the map is provided which guides the segmentation process on an aerial image.

The German Research Institute for Information Processing and Pattern Recognition (FIM) exhibits activities reported by Sties et al. (1977) or Kestner (1980).

1.4 Previous and Current Own Work

The project is based on previous work performed under ERO Contracts and ongoing efforts in the development of geoinformation systems. Kropatsch and Leberl (1981) developed a first concept of a relational digital map data base and showed its applicability in map-guided control data acquisition for digital satellite image rectification.

Leberl and Ranzinger (1982) extended the idea of map-image-correspondence to aerial digital photography. In both approaches, recognition procedures were implemented to identify objects in the imagery with the help of templates taken from the map data base.

The basic image processing algorithms are now implemented on a dedicated device (digital video processor). A comprehensive set of primitive image operations was defined which can be, by means of an interpreting program, combined to perform more complex procedures The idea of the first map data base is currently being extended to develop a geoinformation system (Kainz and Ranzinger, 1983).

1.5 Layers of a Computer Vision System

Problem solving in artificial intelligence involves a multiple-layer structure from the top, where a problem is stated, to the bottom, where circuitry in the computer carries out a sequence of primitive operations fixed by the processor(s) incorporated. Figure 2 gives an idea of this layer structure, where upper layers control lower layers and lower layers serve as tools for operations intended by upper layers. This schematic representation is, of course, not complete, but can be detailed at various levels of complexity.

Basically, top-down concepts or bottom-up-concepts can be constructed. However, top and bottom are ill-defined entities. At each complexity level it may be valid to assume all lower layers to be "black boxes" with an interface only existing to the layer immediately below.

In the problem under consideration, we define the layers "scene analysis" and "map data manipulation" as the bottom and eventually will work upwards keeping in mind that an expert system in computer vision is the final goal. Thus we have to verify that we can make use of tools provided by previous investigations, adapted to a new hardware environment.

2.0 Approaches to Scene Analysis

Change detection in imagery can be approached in different ways depending on the level of data abstraction.

The most simple process involves only the image domain. Two images have first to be registered with respect to their geometries. Leberl and Ranzinger (1982) have shown that modern instrumentation for navigation can give very accurate ancillary data with which a preliminary registration can be accomplished. A fine overlay with sub-pixel accuracy is possible by subsequent digital correlation. Image differencing then yields indicators for changes. The advantage of this method is mainly its easy implementation. A rough sketch of image contents is thus possible. However,

it does not take into account different light conditions and does not yield any clues as to what has actually occured. As a preprocessing step, it may prove nevertheless valuable.

A complex approach takes place on the symbolic level. Here, the image is first segmented into meaningful parts. These parts are then described in a relational structure which also contains shape and grey value properties. These parts are then matched with the symbolic description of the knowledge-base, in this case with the map contents. In many cases where no detailed spatial knowledge is available, the method proves to be feasible. The segmentation process uses only image-inherent information and will thus be rather complicated. But there exists in our context a comprehensive description of what is to be expected in the image which can be used to guide segmentation.

This leads to a third approach which we are taking in this study: The map data base contains positional as well as relational information to make meaningful segmentation possible. The correspondence between map and image which can first coarsely be established by recognition procedures (developed in previous investigations) is stepwise refined by matching objects of the map data base to image features. The segmentation processes can be made considerably complex without becoming untolerably time-consuming, as the areas in the image domain that qualify for inspection are small. The topological relations represented in the map data base can be exploited to make the search for objects goal-oriented.

The spatial description can be exploited to verify recognition by comparing the results obtained to the results expected. Non- verification may point out areas which have undergone changes.

2.1 A Strategy

The strategy to be taken relies on the tools provided.

(a) establish geometric correspondence between map and image
(b) select object from data base
(c) transform object to image data structure
(d) select suitable recognition procedure
(e) recognize object
(f) verify match
(g) if verification successful, mark object as present and continue with (b)
(h) if not verified, mark area as unidentified and continue with (b)

This strategy is terminated if the data base is exhausted or a large number of mismatches indicates a severe error. Result is a list with matched/unmatched object and an image which shows the segmentation results. The interpreter now may enter an interaction with the system to resolve identification problems. Updates are optionally entered to the map data base.

3.0 Tools in Map Data Processing

Based on the experiences gained from a previously used map data base (Leberl and Kropatsch, 1980) the geoinformation system DESBOD is currently under development. An overview of the architecture of DESBOD can be found in figure 3.

The system comprises three principal parts: A data compilation system to digitize spatial data and to assign attributes, a map data base system for management and retrieval, and a data analysis and output system. It is primarily intended to be use for environment - related planning and monitoring and for geoscientific research.

The data structures involved are

- graphic elements and
- thematic elements.

Graphic elements are points, lines and regions which are consist of the graphic primitives "edge" and "node". The graphic elements are on the one hand coordinate-related to represent their spatial locations and on the other hand related to one another by their topologic properties such as adjacency or inclusion.

Thematic elements are assigned to the graphic elements thus giving further descriptions of properties of the real-world-objects represented in the data base. Again,

relations exist between thematic and graphic elements as well as among thematic elements themselves.

For flexible and quick retrieval, most of the relations are stored explicitly so that various data access paths can be selected. Through this construction, it will be possible to extend the system to a general knowledge-base by adding an additional layer which describes, on an abstract level, relations and dependencies of thematic elements to establish a world-model. However, this ambitious extension will involve further research beyond the scope of this study.

The data analysis system as well as the cartographic output system are, at present, of no concern for this work, and will therefore not be described here.

3.1 Connections Between Image Processing and Map Data Processing

The data structures for images and for maps differ because of their acquisition philosophy and the operations intended on them.

Images are stored as matrices and contain at first no explicit information on a structural level, whereas spatial data of maps have the form of vectors associated with location coordinates and can therefore be from the beginning labelled with relational properties. Images are formed "physically" by discretizing a signal which varies over a two-dimensional domain, treating each point uniformly; map

data are digitized "logically" by entering meaningful entities such as lines or boundaries from which the objects can easily be reconstructed.

To use map data in image processing and to incorporate scene analysis results to update map information, these structures have to be adapted to one another.

The procedure of vector-to-raster-conversion is well-known and has been used in previous investigations. Single objects can be retrieved from the map data base and transformed to templates or masks. A more involved procedure has to be applied when a whole raster frame must be filled with labels for different regions which together cover the entire area. Most algorithms have difficulties to preserve geometric properties such as area and adjacency under discrete metrics, especially for small objects.

Raster-to-vector-conversion is, for single objects, also relatively easy to handle. However, the errors occuring during processing (discretisation and curve fitting) do not allow conversions to be strictly reversible. When transforming entire frames to the vector structure an additional problem arises. As the conversion goes from a data structure with low-level implicit relations to high-level explicit relations, these have to be reconstructed. This means that not simply boundaries are of interest, but rather the edges and nodes which separate the different objects. This holds not only for the integration

of analysis results into a particular data base, but also if we try to get symbolic descriptions of the image contents.

4.0 Tools in Image Processing

The image processing system currently in use is DIBAG, supporting research rather than being a production tool. It incorporates actually two more or less equivalent subsets. One subset is designed to work on general-purpose hardware and therefore is basically portable from one computer architecture to another one. The second subset makes use of an interactive image processing workstation and is thus hardware-dependent. However, an image processing language has been defined which allows problem-oriented algorithm formulation. A list of the functions provided now by each of DIBAG's components is given in Appendix A.

We shall outline in the following the hardware environment which is currently used and give a comprehensive overview of the image processing language and its capabilities.

The architecture requires that special applications are implemented at first outside the system itself, but using the conventions regarding data handling. A subroutine library is available which incorporates the basic functions for image access and the user interface. Generally applicable algorithms are incorporated and become standard procedures once they have been proven.

In this study, recognition procedures are of special interest. The original stock of histogram analysis, relaxation and correlation has been extended by a line follower based on gradient magnitude, simultaneous region growing under restrictions and statistical feature-space classification. A warping rectifier was installed to perform the reference grid computation and resampling. Sequences of procedures may now be bound together to yield new functions by writing "macro"-operations in the control language of the computer system.

4.1 Methods for Interactive Image Processing

Experimental work with digital images and graphic information requires a high degree of flexible processing methods. It is necessary to provide an interactive processing facility which has the capability of being fast and transparent. Speed is not only a requirement when running programs in batch mode, but preferably also when it is necessary to establish efficient feed-back cycles during the development phase, especially so for imagery and graphics.

With the large amounts of data involved and the complex and often only statistically describable properties the assessment of the performance of algorithms as well as the debugging of programs is possible often only by "looking" at image-like representations of the processing steps. Thus, a highly interactive modular system of basic functions has

been defined and implemented on which more complex procedures may be built. Main design objectives were speed, transparency and flexibility. The data structure "image", which is the matrix of pixels, either in full resolution (typically eight to 24 bit), or as a binary representation of graphics in general, represents in this context a type of operand for which meaningful extensions of the well-known kinds of mathematical operations have to be defined.

4.2 The Need for an Image Processing Language

Image processing languages are tools to formulate algorithms for image analysis and may be considered as an evolution in high-level programming languages on a problem-oriented level. The interpretation of such languages is presently performed by conventional methods of data processing, but may eventually be implemented in hardware circuitry or firmware.

This section demonstrates how the functionabilities of hardware provided by a typical image processing device with state of the art technology may be exploited by defining a set of elementary operations. First a short taxonomy for computers with parallel processing capabilities is given and the particular architecture of hardware used is described. For a couple of well-known algorithms examples are given how to apply the image processing language for simple construction of procedures.

This approach appears to be untypical - most vendors provide only software for turn-key - systems in specific problem domains with very specialized features. It is in its principles not restricted to a particular hardware system but may be implemented for devices with various architectural characteristics. It represents therefore a step towards higher portability of image processing software.

4.3 A Short Typology of Image Processing Computers

The processing of large amounts of data may be considerably enhanced by performing sets of independent operations in parallel. This requires the design of specialized computer architectures. Following a classification by Karplus and Cohen (1981) or Zakharow (1984) parallelism may be implemented on instruction or data level.

Single instructions on single data streams (SISD) are the well-known capabilities of general-purpose computer systems. The parallelism here is present only insofar as all bits of a data word are processed in one machine cycle. Typically these widths today are 16 or 32 bit.

Multiple instructions on a single data stream are the second possibility to achieve higher speed. Many processes may be partitioned into a number of different simple stages. To process an instruction, fetching of an instruction,

decoding, computation of operand addresses and execution may be implemented in separate hardware units. When processing image data, a typical sequence is transfer of a data element from image memory to processor, a first transformation of the data element by means of a lookup-table, additive, multiplicative or logical combination with another data element, another transformation to modify the result and the transfer of the result back to the image memory. In such cases it is possible to improve the usage of the various units which perform these steps by processing, during each machine cycle, a new element of the instruction or data stream in each unit. This method is known as "pipelining" and results in considerable speeding-up for long uniform input streams. The stream passes through the various processor stages, where each stage performs a particular type of operation on one element while another operation is performed in the previous stage on the next element at the same time.

Vector or array processors (e.g. from Floating Point Systems) but also dedicated image processing computers (some well-known products are from VICOM, Pratt 1981; International Imaging Systems, Adams 1981; or Gould (former DeAnza), Roberts and Shanz 1981) implement this idea. Different complexities of the possible stages provided distinguish the systems. Simple architectures provide only adders and lookup table mechanisms, advanced processors contain multiple multiplying and adding stages with input

data stream widths of 64 bits.

Computer architectures with multiple data streams and single or multiple instruction streams (SIMD or MIMD) contain many processing units. They may be organized as hierarchies, e.g. pyramids, or as matrices (cellular arrays) with interconnections between levels or neighbour processors. Depending on complexity each unit performs one or more simple operations. Their structure is tailored to the immanent two-dimensionality of image data so that performance may be enhanced by some orders of magnitude. The massively parallel processor (MPP, Potter 1983), developed by Goodyear Aerospace and installed at NASA Goddard, for the ground segment processing of Landsat Thematic Mapper data, executes several billion additions per second as compared to several million additions on general purpose computers. This is achieved by a 128 x 128 processor matrix and a sophisticated configurable image data stream assignment and buffering.

It is now the task of software engineering to use the advanced architectures by designing suitable algorithms. Yalamanchili and Aggarwal (1984) propose a systematics by which certain parallel algorithms can be identified and manipulated. Zakharow (1984) gives examples how existing high-level languages may be augmented by constructs (e.g. FORK-JOIN) to indicate parts of codes which may be executed in parallel. The problem of automatic recognition of parallely executable segments of programs by compilers is,

however, not yet solved. Software development in general lags behind the possibilities offered by the advances made in VLSI hardware design and this will certainly hold also for the forseeable future.

4.4 The Image Processor

The device which was user for the development which is described here is the model 6400 image array processor from Gould DeAnza. It is a display system and works with a feedback processor according to pipelining. The processing of the image data as well as the conversion of the digital to analog data to view the image on a monitor is done in so-called video rate.

4.4.1 Features

The IP 6400 in the configuration used here has three image memories with a capacity of 512 x 512 pixels of 8 bit resolution each. An additional memory of the same size and 4 bit depth holds graphics (graphic overlay). Each image memory may also be considered as holding 4 or 8 bit planes to store binary images. The images serve as refresh frame buffers for display on an RGB monitor at a rate of 30 Hz. Contrast enhancement and pseudocolor assignment are provided by sets of lookup tables. By varying the image origin the image planes may be scrolled. The joystick acts as an analog input device and is hardwired to two internal registers which control cursors of programable shapes. An

additional 2000 character memory may be used for annotation purposes.

The device is connected to a VAX 11-750 via the Unibus and thus lies within the virtual address range of the application programs. It may be therefore also considered as a additional dedicated central memory area. The application programs incorporate control modules which access the registers and memories of the image processor, and the dialogue with the user is via a CRT. Computations may be performed by the host CPU or preferrably by the built-in "Digital Video Processor". Through additional hardware, a Matrix 3000 camera, it is also possible to get hardcopies from the images by multiple exposures of monochrome red, green and blue image components through filters on transparency film or through the Polaroid instant process.

4.4.2 The Processor

The Digital Video Processor (DVP) of the DeAnza model IP 6400 consists of two arithmetic/logical units (ASLUs), one 16 bit shifter for result modification and a counter. Two input paths with eight bit width go to each of the units, totalling to an input of 32 bits; two output paths provide 16 bit results. The first processing step is in the "Test"-ALU. Its result may be configured as input to the second step which takes place in the "Operational" ALU. Input can be data from the image memories - they may be

pre-modified by passing them via a lookup-table that represents a mapping of the 256 possible image values - it may be constant values, the result of the Test-ALU operation or the data from a digital camera system.

Together with the results, the Test-ALU generates condition flags which can be used to switch between the two operations presently defined for the Operational ALU. There are instructions of arithmetic type (e.g. constant result, result equal input, sums and differences), or of logical type (bitwise and, or, not etc.). By selecting non-trivial combinations, it is possible to perform rather powerful operations. For example, multiplication can be synthesized by a sequence of additions and shifts.

The results of the ALUs are passed to the shifter and can be shifted or rotated as one 16 bit or as two separate 8 bit numbers. Finally, they are written back into the image memories (Figure 4). In this last step, a variety of write protection facilities is available. First, entire memories can be protected to selct only the ones which shall hold the results. Second, a region stored as a bit mask in the so-called "graphic overlay memory" can be employed to indicate where write-back shall occur. Third, a rectangular region can be defined by two cursors to open the area of the memory for the result. Fourth, write-back can be restricted to bit-planes defined by bit-masks. The write-protection is perhaps the most efficient feature of the processor. Its use enables one to implement also algorithms which are not

strictly parallel and to operate on binary imaages without wasting storage space.

For each operation, a condition can be supplied according to which the counter is incremented. This allows to determine how often the defined operation has yielded a particular type of result.

Neighbourhood operations are based on the scroll-capability. The image P is stored in two copies in two image memories; one memory is then logically shifted with respect to the other one. This shift is defined by the contents of the scroll-registers which point to the addresses from where data transfer starts. If, for example, the register for memory 0 contains (0,0) and the register for memory 1 contains (k,l), the pixels P (i,j) and P (i+k,j+l) will be sent to the processor at the same time. Figure 5 illustrates schematically how an operation is performed.

One operation takes one video time; this is the time required to display one whole image on the monitor or to transfer it to the processor. It amounts to 1/30th of a second. Thus, 30 times up to four 512 x 512 matrices can be treated which gives a theoretical rate of nearly 16 million instructions (counting Test operation and Op-operation).

Before the initialisation of an operation, all relevant registers have to be supplied with values. Depending on the complexity of the desired operation one has to set up to twenty registers and to define a variety of table values. For an application programmer, this is in general too tiresome and requires in-depth knowledge of the processor's structure. The access to the extraordinary power of the device requires evidently the development of a ligh-level software interface.

4.4.3 The Instruction Set

Appendix B lists the current contents of our instruction library. It consists of subroutines written in FORTRAN or in assembly language which can be called by application programs. Alternatively, there is an option to interactively use single operations by means of operation number and parameter input. Results can be immediately checked on the monitor. Operation is possible in dialogue or in batch-mode. For the latter, a file with the necessary inputs (command codes and parameters) must be edited and run through an interpreter program.

The instruction library can thus be seen as an assembly-like programming language for the image processor which works on the data structure image rather than on simple variables.

Parameters for the instructions are operands, modification flags and in all cases the regional write control indicator; it defines the write protection areas mentioned above.

4.5 General Binary Image Operations

Processing of binary images involves operations such as shrinking, skeletonizing or blowing up. These operations are usually performed by considering a 3 x 3 neighbourhood of the pixel. Given a pixel configuration in this neighbourhood, the centre pixel (bit) is set, reset or keeps its value. Therefore it is desirable to get a general representation of the neighbourhood and then define a mapping that transforms the binary image. The representation which we present allows to apply every possible 3 x 3 local binary image operation. We number the surrounding pixels as follows

```
0 1 2
7   3
6 5 4
```

and define a mapping to the integer interval 0,255 by

$$n = \sum_{i0}^{i7} p(i) * 2^{**i} \quad \text{with } p(i) = 0 \text{ or } 1$$

For each configuration, we thus have a unique description n. The general transform now can be represented by two binary vectors of length 256. The first vector gives the new value of the centre pixel for the case that it is set (=1) and for each configuration of the neighbourhood, the second vector describes what shall happen if the centre pixel is zero. For example, if we want to delete pixels that have no neighbours and at the same time fill gaps when all surrounding pixels are set, the vectors are v1 = (0,1,.....,1) and v2 = (0,.....,0,1).

To implement this idea with the DVP, we scroll the mask eight times and copy it to ascending bit planes of an image memory with the instruction DVPMCY. The resulting image defines the neighbourhood (in some sense textural) information.

The transition of the original to the new mask requires two more steps: The regional write enable bit is set for locations where the original has "one"-values, the vector v1 is loaded into the transformation table for the image memory where the neighbourhood image resides and this image is then copied to the output bit plane with the table enabled. The same is done for original zero pixels and the vector v2, respectively. The whole procedure takes ten video times. Its versatility lies in the fact that the vectors are stored on files which may simply be edited and kept for various operations on binary images and are eventually read by the general procedure described above.

4.6 Thresholding and Histogramming

The thresholding operation is performed on the DVP by employing the lookup-table transformation function which can be applied to each pixel as it enters the processing pipe (For technical reasons, this transformation is not directly considered as being part of the DVP pipeline, as all the logic necessary to perform it resided on the memory board and not on the DVP board; in the advanced architecture of the IP8500, this distinction is still more obvious). Setting this lookup table of the appropriate channel to one in the threshold range and zero elsewhere and enabling the transformation for data coming from image memory results in the correct threshold values to arrive at the DVP.

The ALU operation code is set to pass data straight through without any modification. The output may be written to any bit plane of the graphic overlay, so a shift operation has to be performed before data are written back to memory. Regional write control is permissable when using this operation, DVPTHR. It is implemented as a function which passes as a result the area of the binary threshold image back to the calling program. This area is accumulated automatically by setting the appropriate code for the counter. DVPTHR takes one video cycle for execution.

The histogram of an image is computed by stepping through the entire value with the values in the image. The computer is incremented when the equal condition is fulfilled. At

the end of each step, the count register is read out and transfered to the corresponding entry in the histogram array. It has to be noted that this process is not particularly well-performing. There is hardware available - for example in the IP8500 - which captures a histogram much faster by providing multiple counters, so that only one to four video cycles are consumed. It turns out that in the current configuration it is more convenient - from the point of performance - to read out the image memory in large buffers to the host's main memory and scan the array conventionally, using each gray value as an index into the histogram array and incrementing the corresponding location, just as the well-known algorithm works.

In the early version used on the PDP-11, additional overhead was imposed by the restriction that only 15 bits were provided for positive integers, on the 32-bit machine it is no more necessary to take special precautions for arithmetic overflow when a few values dominate in the image.

For two-dimensional histograms which are useful in multispectral images or when computing special types of texture parameters such as co-occurence matrices, the DVP approach has to be ruled out as it is too slow. Such statistics are gathered in the usual way, only using the image memories as additional storage to the host's central memory.

To complete the set of functions provided for counting operations, let us note that there exists an operation similar to DVPTHR which is called DVPNPX and differs only in that there are no values written back to the graphic overlay; only the count of pixels is returned.

5.0 The Implementation of Some Basic Recognition Algorithms

The three recognition algorithms used in previous work (Kropatsch and Leberl, 1981 and Leberl and Ranzinger, 1982) were transported to the now image processing environment. This was decided to be necessary as well for the sake of performance and transparency as also for the uniformity of appearance for other in-house available software. This section recollects in short the background of the algorithms and describes the mechanisms used for the reimplementation.

5.1 The Background of the Recognition Procedures

The inputs to the recognition procedures for areal features to be discussed here consist of

(a) a binary mask of the map feature after extraction, transformation and vector-to-raster conversion, on the image processor; this binary mask is read into bit plane 0 of the graphic overlay.

(b) the section of the digital image which contains the object to be recognized; this image (it is assumed to be monochrome with eight bits of radiometric resolution) is kept in image memory 0.

(c) an estimate of the distance between the position of the map feature and the homologue image object, here denoted as "d"; this distance is used to restrict the search area.

(d) an estimate for the distortion of the map feature with respect to the image, expressed as a percentage of the area of the binary mask.

Figure 6 depicts the initial situation for each of the recognition algorithms.

5.2 The Algorithm THRESH

The rationale behind the basic algorithm using thresholding is that an object which has a restricted gray-value range may be segmented from its surroundings if one is able to use the mask of the map feature to approximate the actual objects area. This area corresponds to a number of pixels in the histogram of area surrounding the mask. To account for errors in the approximation of d, the mask is extended by adding a band of pixels of the widths d to the initial projection (Figure 7). Considering

the histogram taken under this extended mask, it is attempted to establish a lower limit and an upper limit on the gray value axis in such a way that the length of the interval - which corresponds to the intensity range of the object - is kept minimal and the sum of the histogram counts exceeds the estimation for the object's area (Figure 8).

Thus the algorithm consists of the following parts

- enlarge the mask
- take the histogram
- establish upper and lower bounds
- perform a threshold
- smooth the result

Note that the last part is necessary to account for the problem that the mask after thresholding can be fragmented, contain holes and there may be pixels outside the actual area which happen to lie in the computed gray value range, but are not part of the object.

The following descriptions show how the parallel processor is used to implement some of the steps of the algorithm on the basis of the command language for image processing. When computing the enlargement, for each pixel it is determined how many pixels are adjacent to it under some metric.

Assume that the mask is stored in bit plane b of the graphic overlay. To enable the writeback of the processor only in areas which are covered by this mask, the write control is set to this bit plane by issuing SETRWC(2**b,1,1). (For a full explanation of all parameters see the Appendix B). A target image channel is cleared first; then the mask is incrementally scrolled to all positions which are connected to the central pixels and each time the target channel is incremented by one where pixels of the scrolled mask are aligned to the (unscrolled) target. The command NBCNT does all this work and returns as a result an image in which each pixel contains the count of neighbours. Actually, NBCNT is more general: first, it can handle greater neighbourhoods, and second, it handles various metrics, including

```
D((IX,IY), (JX,JY)) = MAX(ABS(IX-JX), ABS(IY,JY))

D((IX,IY), (JX,JY)) = ABS(IX-JX) + ABS(IY-JY)
```

and an approximation of a Euclidean metric, thereby forming a digital disk-shaped neighbourhood. For a detailed discussion of digital disks, see Kim (1984).

It is evident that pixels which do not lie under the original mask and for which the corresponding pixel has a non-zero count of neighbours will belong to the enlarged mask. This enlarged mask can therefore be found out simply by thresholding the target channel in the range from one to

the maximum possible count of neighbours. Two variants are possible to use NBCNT for the enlargement of the mask. In the first one, the original mask is shifted to every legal position in a window of size 2*d+1. Let d be 7 and the metric the sum of the absolute coordinate differences. Then 225 different positions have to be taken. The second variant computes intermediate results: the mask is extended stepwise, each time using only a 3x3 window (9 positions in our example) and thresholding the image back to the bit plane where the mask from the previous step resided. For d = 7, the entire operation takes then only 63 videocycles for the computation of neighbour counts and 7 thresholds, one cycle for each of them. Both approaches are equivalent only for the 4-neighbourhood and 8-neighbourhood metrics. The approximation of the Euclidian distance gives a correct result in the first case, in the second case the result will be equal to that in the 8-neighbourhood, as rounding forces the 3x3 digital disk to indicate the same displacements (or scroll positions); this error propagates then though all the steps until d is reached.

The histogram now is computed for the region of the image covered under the mask. As pointed out before, this is done more efficiently if transfers to the host's memory are used instead of DVP operations, especially if the window in which the enlarged mask lies is known and only those lines have to be read from the image refresh store.

As the algorithm to determine upper and lower bounds for the object's gray value range works on the histogram array, no modification with respect to the DVP code had to be made.

The threshold is performed again with the operation DVPTHR, resulting in a segmentation of the image into object and non-object areas with noise occuring for the reasons already mentioned above.

To account for that noise, several approaches were designed. Starting from the assumption that the object area is the largest connected area of pixels in the new binary image which indicates the position of the objects, a formal deletion process may be used: In a first step, all isolated pixels are removed and small holes are filled in the larger area. The mechanism used here is the binary image operation described previously. Larger artefacts can be accounted for by setting the transformation vectors to the desired configurations of pixels in the neighbourhood, for addition as well as for deletion.

Another way to clean the threshold result is to use region growing. Usually here one starts from a seed pixel or set of seed pixels which are set by the operator. Such manual interaction is not desirable at an early stage of processing, and an automatic approach to find the seed area must be provided. This is achieved by again using the "largest area" assumption. It is possible by using the operation NBCNT, to generate again a neighbour-count array

for the area within the window inspected. The values of this array which are maximal indicate where the greatest density of the mask pixels is found. This maximum area is taken as seed area for the region growing.

Growing the region follows the same principle as counting neighbours. The seed region is enlarged stepwise by adding a band of pixels around its border. The enlarged seed and the binary image are then subjected to a logical "and" operation (DVPMND). This prevents the seed from bridging the gaps around the region to which region growing is applied. The result of the "and" operation is defined as the new seed region, and the process is applied recursively until no change in the area is observed. The previous binary image is replaced by the new result.

To fill the gaps which might exist within the object, the same region-growing may be applied to the complement of the mask found so far in the same way. Complementing the second result again gives a connected area which hypothesizes the situation of the object on the image.

It is important to see where and why the THRESH procedure has to fail. The one case is that of lacking contrast or a highly texturized object. The determination of upper and lower gray value bounds will then give arbitrary indicators, and after thresholding one may well end up with a mask which covers the window randomly. The second case is found when the ratio between the area of the enlarged mask and the area

of the original mask is high. We found that a ratio of 1.5, under average contrast, is enough to include so much background that an accurate value range cannot be established anymore. The ratio is dependent on the size of the object and its form, as well as on the value of d. The smaller the object and the higher the ratio of the sides of a circumscribed rectangle - not necessarily the one parallel to the coordinate axes - and the higher the value of d, the more background has to be included, making recognition less likely.

5.3 The Algorithm ADAPT

The adaptive method implemented by the algorithm ADAPT is sequential in its nature and therefore is the least suitable for a reformulation with the DVP command set. The underlying idea is as follows: Start from the projected mask and estimate upper and lower bounds for the grey value range of the object from the histogram under the mask, using the same procedure as in the algorithm THRESH. These values are then used to move the area covered by tracking the border of the mask, omitting border pixels which lie outside the interval and adding pixels with values between the bounds.

The important issue here is to keep the original topology; that means that the modification must not break the mask into several parts or, in other words, to maintain the connectivity properties. To this end, the general

neighbourhood description is computed and configurations are identified in which deletion of the central pixel is permissible. As the neighbourhood description exists for mask as well as for non-mask pixels, it is also possible to simply classify the neighbourhoods into border and non-border pixels.

Taken together, three cases may be considered: outer border pixels which are non-mask pixels where pixels may be added if they fulfil the interval condition; inner border pixels which may be omitted from the mask; and inner border pixels whose omission would cause the mask to break up. In each case, the chosen metric influences of course the pixel's "border" or "simple" property where "simple" is the term for a pixel which has a connected neighbourhood under the metric. The various classifications of pixels are shown in figure 9.

The determination of a pixel's properties is recursive in nature; that is, it cannot be applied globally before tracking the border, but instead has to take into consideration also the outcoming of the previous step, as, for example, the addition of an outer border pixel may well change the border property of an adjacent pixel. The adaption thus makes only use of the image memories as extended storage space for the program.

The classification makes use only of the lookup tables in which the various properties are held. Cycling around the mask is performed d times, and the result hypothesizes again the position of the object. It may be said in favour of the adaptive algorithm that no post-processing has to be applied to eliminate noise, as it cannot occur.

The probability of failure of ADAPT is high if the initial projection does not cover a major part of the object to be recognized. This happens if the extent of the object in one direction (measured as the side length of the bounding rectangle's smaller side) is near to the half of the distance d. If the object's background is uniform, the upper and lower bounds will then be the ones for the background, and the mask will inevitably move into the wrong direction, away from the object.

5.4 The Algorithm SHIFT

This algorithm determines a displacement vector between the original projection and some location where a maximum of a similarity measure is found. The similarity measure is here a normalized absolute difference between the mean gray value of the area under the mask m(M) and the mean gray value of the rectangular window forming the search area m(S).

$$c = \text{abs}\,(m(M)-m(S)) \;/\; s(S)$$

where the denominator s(S) denotes the standard deviation of the image function in the search area. It has been shown by

Kropatsch (1981) that a suitable choice of a constant, depending only on the areas of mask and window makes c a product moment correlation coefficient between a binary image and a gray image.

The entire process consists of shifting the mask within a window which is the mask's window measured parallel to the coordinate axes enlarged by the distance d on either side and computing the similarity measure c. Only m(M) depends on the location of the mask, the other values are global for the search and therefore have to be computed only once. The mean value is computed by using DVPSUM under the control of the mask.

DVPSUM looks at the image as a set of bit planes and computes the area of each plane by using both of the ALUs of the DVP. In the test ALU, a bit plane mask containig a one only for the desired bit plane is used to filter out only the bit plane considered using the logical 'and' opcode. The result is passed on to the operational ALU which compares the values to a constant which is again the same bit plane mask. The counter is enabled for incrementation in case of equality. Stepping through this operation once for each bit plane and using the digit values of each bit plane 2**i as the factors for the areas, yields the sum in eight video cycles. When finding the maximum value of c, the corresponding location gives the best fit of the mask to the image function. The maximisation finds, in other words, the area of the greatest contrast to the background in the

search area. The mask is brought to the new location and hypothesizes the position of the object.

Again the contrast of object and background is the major performance criterion. Weak contrast of object and background results in a relatively flat array of similarity values c, and there may be no predominant maximum to be found. There is also the disadvantage that no form change is made to the mask, so that distortions caused by the projection are not accounted for. SHIFT is thus useful to determine the position, but not the form of the object.

5.5 Combinations

It was investigated how the different merits of the recognition procedures could be synergetically used to give a more precise hypothesis about an object. As SHIFT is the most robust one, it may be used to restrict the search area for one of the other algorithms to a place nearer to the object. Then the problem of getting too much background into the histogram and the potential failure to determine upper and lower bounds for the gray value range of the object is avoided. This approach turned out to be feasible in some of the experiments, but further evaluation is still needed.

6.0 Test Data and Experiments

The aerial imagery selected for test purposes consists of a multitemporal series of four overflights which cover a period from June, 1968 to May, 1982. Scales range from 1:9000 to 1:30000, thus representing different degrees of detail. The imaged area lies south of Graz and was also used in previous studies. There is no significant terrain relief so that problems with geometry should be minimal. The four images wer digitized on an Optronics scanner with a resolution of 100 micrometers, and an area of 2048 x 2048 pixels was in each case cut out (window 101, 101, 2048, 2048). The images are shown in figure 10.

The photos document urban growth with new infrastructure (motorway), industrial settlements and suburban housing. The river Mur serves as an invariant backbone as well as some major roads in the area. Agricultural land use and forest are other dominant components. Thus, various tests can be carried out on features with different characteristics.

For the experiments, two of the four images were selected because of their geometric similarity. Preliminary experiments had shown that there are great difficulties when dealing with imagery which has a significantly scale: texture of an object becomes more apparent and tends to disturb the behaviour of the recognition algorithms. The two images selected were the ones acquired in 1975 (named

GRAZ75) and in 1979 (named GRAZ79 for short).

First procedures had to be found to transform the imagery in such a way that ist could be displayed on a variety of output devices. Two of them are of particular interest: the video display on the monitor of the image processing system and the electrostatic plotter. (A third device, the color graphics recorder, exposes film to a monochrome video image on the built-in monitor and behaves similar to the image processing display, so no changes have to be made here). The electrostatic plotter imposes two restrictions: First, it works in subtractive mode as opposed to the additive mode on the video displays. Second, it has far less resolution (about eight gray values can be shown as opposed to the dynamic range of 255 gray values of the video display). To take this into account, the image in the refresh memory must be mapped to the dynamic range of the output device. The first step is a linear contrast stretch for which the lower and upper bounds are computed on the DVP from the statistics of the image. It turned out that bounds set at the first and 99th percentile of the cumulative histogram were best suited for the type of imagery under investigation. The program ENHANC implements this idea. The second step is the reduction of the resolution to three bits per pixel. A lookup table with eight steps in the interval 0,255 is set and the image is copied via this table to store the values permanently. The program REDRES is used in this stage. Figures 11 and 12 show what can be achieved

by this process.

Several maps were digitized with the manual digitizing function provided by DESBOD. Figure 13 shows a part of a map made for the airport scene of figure 12; figures 16 and 17 illustrate linear structures (roads and railroads) as they are plotted by the DESBOD map output subsystem and, after a polynomial transform - for which the coefficients were computed from control points identified on the map and in the image - and vector-to-raster conversion, plotted in the 512 x 512 resolution of the image processor. These maps will be used in the future when more and finer recognition algorithms become available. One of the anticipated ways is outlined in figures 14 and 15. Figure 14 shows the map of figure 13 superimposed on the airport scene. Figure 15 gives an example of one of the edge operators (ROBERTS) now available. Other operators implemented include SOBEL and LAPLACE, and the image processing language may be used to compute also more special derivatives, e.g. direction-sensitive ones. These edge images may serve as a basis for a map-guided line follower based on gradient magnitude, as Pessl (1983) has shown.

A third excerpt from the map contains a set of prominent areal features, such as buildings, fields etc. Some of them were used to test the re-implemented recognition procedures and to show their limits with respect to the detection of changes in the multitemporal set of images.

The second experiment tested in which way changes could be located by image-to-image comparison alone. From each of the two images, control points were determined interactively using the program SELCP. Using this information, the distortion of GRAZ79 was determined with respect to GRAZ75 and GRAZ79 was resampled to the geometry of GRAZ75 using the program modules from the system RECTIF (Diarra, 1983). Figure 18 shows the subscene from GRAZ75, figure 19 a roughly corresponding subscene of GRAZ79, and figure 20 the result of the rectification process. Image differencing was then attempted between the two images, but turned out to be of little value for the problem under consideration. There are several reasons why this is so. The images differ with respect to lighting conditions and overall image intensity; this results in small differences computed even for objects which have not changed. There may be differences in the gray values when the color of an object changes, but not the form; this is the case especially for agricultural areas which are usually planted with different crops in different years (see figures 21a and 21b). The resampling process is not accurate enough to achieve subpixel accuracy; therefore there will be always high differences along the borders of high-contrast areas. The attempt to establish initial hypotheses about change areas by image differencing had therefore to be abandoned.

The next set of experiments was carried out on a subscene contained in both of the images where the behaviour of the recognition algorithms could be tested for a case where an object (a factory building) had not changed within the four years between the acquisition of the two images and another object (a park) had been replaced by a new building. Figures 22a, 22b and 23 illustrate this situation.

All three recognition algorithms were applied to both of the areas. The distance d which accounts for inaccuracies of the projections was set to 5 in all cases, and the area factor which is used when establishing lower and upper bounds of the gray value ranges (see figure 8) was set, after some testing, to 0.9. The results are given in figures 25 and 26.

The SHIFT algorithm was able to locate the unchanged object in both the 1975 and 1979 image with similarity measures of 0.4494 and 0.2387, respectively. The lower value for GRAZ79 may be explained by the fact that in the neighbourhood of the object the image intensity increased (compare figures 18 and 19). The similarity measure for the area in which a change had occured was considerably lower. For GRAZ75, the displacement vector was (-2,-3), and well within the search area, whereas the displacement of (+5,+5) for GRAZ79 shows that the search area had been exhausted and an optimum was not reached. A further extension of the search area did not lead to a significant increase of the similarity function, indicating that no object of the sort

described by the map was found.

The procedure based on thresholding proved to be well-suited for the recognition of an object which is homogeneous in appearance, such as the factory building. In both the 1975 and the 1979 images, the results were appropriate when compared with the projected mask of the mask feature (shifted by the amount indicated by SHIFT). For the more textured area of the park, there is already some difficulty to perform verification of the recognition results: some patches inside the region were not filled by the smoothing operation applied (binary mask transformation) and the comparison with the ideal mask gave weaker evidence of the existence of the object. When the park was replaced by a building with more homogeneous appearance, THRESH was able to detect the new structure (figure 26c) and the verification attempt clearly failed indicating the change.

ADAPT exhibited the most serious problems. It has a strong tendency to keep the form of the old mask if there are gray values near to those of the object itself in its neighbourhood. This results in artefacts such as a rugged border, such as is seen on figures 25d and 26b. This noise makes verification difficult. Figure 26d shows that the adaption process did not manage the transition from the old form (the park mask) to the new form (the building), presumably because the gray value bounds were estimated inadequately.

7.0 Conclusions

The problem of knowledge-based change detection on aerial imagery for the purpose of map updating poses a series of interesting and challenging problems. Thorough studies can be performed only if a testbed is available which has enough flexibility to allow experimentation with different approaches. The methods which have to be provided come from a wide range of sciences. Digital cartography allows the computer-based description of real world objects - the geo-information system DESBOD was developed to accomodate one type of relevant knowledge for the interpretation of aerial photography. Other projects currently carried out at the institute will lead to tool for maintainance of map-image correspondence via analytical camera models. Their results should be in due course incorporated to advance further studies into more rigid correspondence issues.

It has been shown that in image processing a variety of basic procedures is necessary to build advanced recognition algorithms. The image processing language introduced here has proved to be a valuable tool to do that. It has been demonstrated how algorithms proposed in previous studies can be implemented using this language and other now existing functions in the DIBAG software system.

The experiments carried out clearly indicate that the way to an automated system for photo interpretation is long and that substantial progress still has to be made. This includes issues like texture processing, spatial constraints, extraction of symbolic information and feature matching on the symbolic level, and the augmentation of systems to three-dimensional representations to build more adequate models of the world.

R E F E R E N C E S

ADAMS J.R. (1981): Display or Processor? Proc. SPIE, Vol.301.

BALLARD D.H, BROWN C.M. (1982): Computer Vision. Prentice-Hall, Englewood Cliffs, N.Y. (USA).

BARROW H.G. et al. (1977): Experiments in Map-Guided Photo-Interpretation. Proc. 5th Intern. Joint Conference on Artificial Intelligence, MIT, Cambridge, Mass., August 1977.

DIARRA G. (1983): RECTIF - A Digital Image Rectification System. DIBAG Report No.10, Forschungszentrum Graz.

FISCHLER M., TENENBAUM J., WOLF H. (1979): Detection of Roads and Linear Structures in Digital Images. AI-Laboratory, Stanford, TR 200, Menlo Park, Cal.

HAVENS W., MACKWORTH A. (1983): Representing Knowledge of the Visual World. IEEE-Computer, 10/83.

KARPLUS W.J., COHEN D. (1981): Architectural and Software Issues in the Design and Application of Peripheral Array Processors. IEEE-Computer, Vol.14, No.9.

KESTNER W. (1980): Considerations about Knowledge-Based Image Interpretation. Proc. IAPR, Miami.

KIM CH.E. (1984): Digital Disks. IEEE Trans. Pattern Analysis and Machine Intelligence, Vol.PAMI-6, No.3, pp.372-374.

KROPATSCH W., LEBERL F.W. (1981): Automated Registration of Scanned Satellite Imagery with a Digital Map Data Base. DIBAG Report No.1, Forschungszentrum Graz.

LANTZ K.A., BROWN C.M., BALLARD D.H. (1978): Model-Driven Vision Using Procedure Description: Motivation and Application to Photointerpretation and Medical Diagnosis. Proc. SPIE, Vol.155, pp.144-157.

LEBERL F., KROPATSCH W. (1980): Experiments with Automatic Feature Analysis Using Maps and Images. Pres. Paper, 14. Kongreß der Internationalen Gesellschaft für Photogrammetrie, Hamburg, BRD, 13. - 24. Juli 1980.

LEBERL F., RANZINGER H. (1982): Registration of Digitized Aerial Photography with a Digital Map Data Base. DIBAG Report No.5, Forschungszentrum Graz.

McCALLA G., CERCONE N. (1983): Approaches to Knowledge Representation. IEEE-Computer, 10/83.

McKEOWN D., J.L. DENLINGER (1982): Graphical Tools for Interactive Image Interpretation. Computer Graphics, Vol.16, No.3, pp.189-198.

McKEOWN D. (1982): Concept Maps. Proc. DARPA IUS Workshop, Stanford, Sept. 1982.

NAU D.S. (1983): Expert Computer Systems. IEEE-Computer, 2/83.

PESSL P. (1983): Kenntnisgest}tzte Liniensuche in digitalen Bildern. Diplomarbeit, Technische Universit{t Graz, DIBAG Report No.7, Forschungszentrum Graz.

POTTER J.L. (1983): Image Processing on the Massively Parallel Processor. IEEE-Computer, Vol.16, No.1, pp.62-67.

PRATT W. (1981): System Architecture of VICOM Digital Image Processor. Proc. SPIE, Vol.301.

ROBERTS L.H., SHANZ M. (1981): Processing Display Systems Architectures. Proc. SPIE, Vol.301.

STIES M., SANYAL B., LEIST K. (1977): Organisation of Object Data for an Image Information System. FIM-Institute, Karlsruhe, FRG.

TENENBAUM M., M. FISCHLER, H. WOLF (1978): A Scene Analysis Approach to Remote Sensing. Technical Note 173, Artificial Intelligence Center, Stanford Research Institute, Menlo Park, Kalifornien.

YALAMANCHILI S.,AGGARWAL J.K. (1984): Formulation of Parallel Image Processing Tasks. Pattern Recognition Letters 2, pp.261-270.

ZAKHAROV V. (1984): Parallelism and Array Processing. IEEE Trans. on Computers, Vol.C-33, pp.45-77.

APPENDIX A

THE IMAGE PROCESSING ROUTINES

D E A N Z A - Instruction Set

Image Transfer

SAVPIC	save image memories to disc file
SHOPIC	get image from disc file
SHO256	display four different images in 256**2 windows

Image Processing

ADATHR	adaptive thresholding
AUTCOR	autocorrelation function via DEANZA
AUTCORB	autocorrelation function via DEANZA on binary images
AUTOCR	autocorrelation under a 256x256 (max) cursor rectangle
AVERAGE	image smoothing via CPU
BASRLF	biased simple edge operator
CONTOURS	compute contour lines
CONVLV	convolution with cos-exp or own mask
DIDXY	simple edge operator
DIFVEC	difference vector image from max.5 images
DIST	mask distance transform
DVP	DVP program interpreter
LAPLAC	Laplace edge operator
MEDIAN	median approximation under 3x3 mask
MINMAX	minimum and maximum under nxn mask
PCT	principal component transformation - compute
PCT1	principal component transformation - transform
RATIO	ratio image
REGAVG	regional averige for texture parameter acquisition
ROBERT	Roberts edge operator
SOBEL	Sobel edge operator
THRESH	interactive threshold determination
VECLEN	create vector length image

Binary Image Processing

BINRNK	rank operator on binary image
BLOSHR	blow/shrink rank operator
CONTRACK	track contour of mask
MASKCL	clean binary mask
MASKOP	set/clear pixels in binary mask according to neighbors
MASKWD	calculate DIBAG window of binary mask
PCTBIN	PC-trafo for masks
SHOBIT	show binary image transformation table
SKELET	skeletonisation

Memory Manipulation

FLIP	mirror or rotate image channel
SPLIT	copy or mirror half image

Image Enhancement

ENHANC	image enhancement via LUT; lin.ramp from 1-99%
IHSRGB	intensity-saturation-hue to RGB
IHSRGBHR	intensity-saturation-hue to RGB; high resolution
IP5MCC	density slice contour lines
IP5UNI	histogram equalisation
JP5ITT	manipulate lookup-tables with joystick
MANITT	special functions for rendition on screen
PSEUDO	interactive pseudo coloring module
REDRES	change image resolution
SETITT	manipulate ITTs or LUTs
SETPCL	assignment of selected colours to arbitrary pixelvalues
SFG	set SFG parameters
SHSPLT	set and manipulate splitscreen coordinates
WIND16	manipulate ITTs for 16 bit image data

Histograms and Statistics

HIST0	calculate print and plot a histogram
HIST2	two dimensional histogram
LINHIS	generate line histograms
PIXVAL	interactive pixel value determination
ROBUST	robust statistic estimates from image
SNSHIS	separate histograms for each Landsat sensor
THRLIM	calculate limit under which p percent of pixels lie

Zoom and Scroll

INTPZM	zoom with software interpolation
JSCRZM	scroll and zoom image
JSCZM1	scroll and zoom image 1024*1024
JSELZM	zoom image
LDSCR	scroll three images simultaneously in SFG windows
SMALL	reduce 512x512 image to 256x256 in upper left quarter

Graphic Overlay

DEFTA	define training area (contains features of REGDEF and SETOV)
PLTMAP	superpose map (vector) data
REGDEF	define region with joystick and vector fill
SETIT4	manipulate ITT for graphic overlay
SETOV	set pixels in G/O with joystick
TICKS	plot tick grid or various marks

Image Annotation

EDITAN	edit annotation overlay
IP5ANW	write annotation
SHOWAN	get annotation from disc file
SHOWPG	load the A/O from a text file
SAVEAN	save annotation to disc file

Miscellaneous

ANZDEZ	print image window values defined by cursors
BWLOOP	loop through black/white rendition of channels
GENPIC	generate .PIC file with 0-values
IMLOOP	show animated 'film'
PLOTCP	plot control points
SELCP	select control points
SKALA	show scale marks on reference scales
WRSCAL	write reference scale to image file or channel

Classification

NOTE : the following programs should be run under the command procedure MAINCLASS

ANACLA	analyze classification statistics; print value ranges of gray values for each image
ANAHIS	analyze histogram to show multiple classifications
CLADIS	compute class.distances per image on .STA file
CLUSTA	cov.matrix,inverse and determinant; corr-matrix; eigenvectors of the cov-mat.for a set of train-pixels
CMPCLA	comparison of classification results
JMDIST	distances of clusters
MAXCLA	reclassify image according to most frequent class in the neighbourhood of each pixel
MAXLIK	maximum likelihood classification
MINDIS	minimum distance classification
PAREPI	parallel epiped classification
PIXLST	list of pixel coordinates which are marked in G/0
PIXSTA	compile statistics for training areas
PLTFEA	plot of feature vectors
PLTMAP	draw situation over img
PXVLST	get feature values for pixels in training areas
RECLAS	reclassify multiply classified pixels
SCATTER	scatter plot of training pixels
SHOCLA	show pixels with cursor defined values in two images
SHOCLU	interactive classification
TESTMD	a priori test minimum distance classification
TESTML	a priori test max.likelihood classification
TESTPE	a priori test parallel epiped classification

Object Recognition

DARECA	ADAPT algorithm
DARECS	SHIFT algorithm
DARECT	THRESH algorithm

Versatec Interface

ANZBIN	output of binary mask of G/0 to VERSATEC or .BIN file
ANZCOL	output of image with gray scale software to VERSATEC
BINOUTPRI	genererate plotfile for PRISM dot matrix printer

Test and Demonstration

COLTAB	color table for color selection
DEMBOX	demonstrate rectangle generation
DEMCUR	demonstrate cursors
DEMZOOM	demonstrate zooming (512 x 512) to (1 x 1)
POLYDG	demonstrate vector plotting
SPIRSC	scroll image channels in spirals
TESTIM	various test images and patterns

Management

CREIP5	connect VAX to DEANZA
DELIP5	disconnect VAX from DEANZA
DEADMP	dump of DeAnza registers octal
DVPTST	define an instruction for the DVP
SETITT	manipulate ITTs or LUTs
SETIT4	manipulate ITT for graphic overlay
STATUS	show status of deAnza environment
SWITCH	switchboard utility
SYSINT	system initialisation

D I B A G - Instruction Set

Note : Instruction marked with an asterisk (*) require special hardware devices; instructions marked with a plus (+) sign are computer installation dependent

Read/write of Magnetic Tapes with Various Formats Data Reformatting

CT256		computed tomography
DEZDSK		read list with decimal grey values
LDSAT		LANDSAT MSS - NASA
OPTIN		read Optronics - scanner image data
OPTOUT		write image data for output on Optronics filmwriter
SAR		SAR-580 (airborne radar)
SARTEST		ancillary information SAR-580
SEASAT		SEASAT JPL - format
TELSAT		LANDSAT MSS Telespazio - format
TELEHD		ancillary information LANDSAT - Telespazio - format
TELRBV		LANDSAT RBV Telespazio - format

Image Rendition

BDDEZ		print image grey values as decimal data matrix
BDGRAY	*	image hardcopy on electrostatic plotter
PLOTFO	*	plot contour lines
PRINT		print image on lineprinter with 8 grey levels and automatic histogram equalisation

Image Manipulation

CHEOPS		image pyramid generation
COPYBD		copy (sub)image considering optionally given lookup-table and binary mask
DIRECT		geometric transformation (magnification, reduction, rotation)
GRKEIL		insert grey value reference scale into image
LADBGR		create image file with constant pixel value
NOISE		impose synthetic noise onto image

Image Statistics

GENHIS		generate histogram and store on work-file
HIST2		generate two-dimensional histogram and store as image
LIHIS		read histogram from image file to work-file
MINMAXBD		compute minimum and maximum value in image
PLOHIS	*	plot histogram
PRIHIS		print histogram
SPAHIS		column histogram of binary image
STOHIS		store histogram in image file
ZEIHIS		line histogram of binary image

Gray Scale Modification (Lookup-Tables)

HISLOK	generate table for equalisation according to given distribution function
LOK	manipulate lookup-table
NEGATE	invert lookup-table
SETLOK	set linear table

Spatial Filters

LAPABS	Laplace - absolute value sum
LAPLAC	Laplace
LAPPOS	Laplace - absolute value
MASK33	general 3x3 - filter with user-defined weights
RANGOP	rank operator (e.g. median, minimum, maximum)
ROBABS	Roberts - absolute value
ROBERT	Roberts
ROBMAX	Roberts - maximum
SMOOTH	averaging
SOBABS	Sobel - absolute value
SOBEL	Sobel
SOBMAX	Sobel - maximum

Image Combinations

ADIMAG	linear combination f(G,H) = a.G +/- b.H
DIST	distance image (values represent distances from mask)
RATIO	ratio image
THRESHOLD	image thresholding
VECLEN	vector length image

Processing of Binary Images and Masks

BINAREA	compute area and centre of gravity
BINCORR	correlate masks
BINDRU	print mask
BINEX	disable processing of mask
BINFIL	fill interior of area given by mask
BININ	enable processing of mask
BINOP	logical combinations of masks
BINSHI	shift mask within its window
LIBIN	load mask from image file to work-file
LIMAS	load mask from mask file to work-file
MASKBD	set image pixels to constant value at locations defined by mask (choropleth generation)
NOTBIN	complement of mask
REGION	region growing
STOBIN	store mask from work-file on image file
STOMAS	store mask from work-file to mask file

Line Following

PASCOO	*	interactive input of line transition points
LINSEA		line following - forward search
TRACK		line following - backward search

Vector-to-Raster-Conversion

POLBIN		rastering of polygon given by vectors, result is binary image, optionally with polygon fill
POLFIT		smoothe polygons
POLGEN		generate polygon file from level contours
POLINF		print polygon file information
POLPLO	*	plot polygons

Geometric Rectification

GRIGEN		compute deformation description grid from control point data with polynomial warp functions
GRIANA		control print of grid data
PLOKNO	*	control plot of grid data
RESAMP		image resampling
CORREL	*	digital image correlation

Digital Terrain Model (GTM) Processing

BDGTM		convert DIBAG-image to GTM-format
GTMBD		convert GTM-format to DIBAG-image
OBERFL		generate surface file for perspective view rendition
D3DISP	*	synthetic perspective view of terrain data with different illumination models optionally generation of distortion information
PQPARA		compute local normal vectors
SLOPE		compute slope and exposition of surface points
VISBIN		generate mask with visibility information
ILLUIM		generate image with synthetic illumination

Interface to DeAnza image display and image array processor

ANZABD	*	read image from display memory and store in DIBAG-Format on disk
ANZLOK	*	read lookup-table from table memory and store in DIBAG-work-file
ANZMAS	*	read binary image from graphics memory and store in DIBAG-work-file
BDANZA	*	display subimage on monitor (8 bit quantisation)
DISPLAY	*	display subimage on monitor (16 bit quantisation)
LOKANZ	*	transfer lookup-table from DIBAG-work-file to table-memory
MASANZ	*	transfer binary image from DIBAG-work-file to graphics memory

ASN	+	assignment of input/output devices
BDINFO	+	information about existing images
BKW	+	definition of actual image name and window for subsequent processing steps
COM	+	insert comments into work protocol
END	+	end of DIBAG session
EXIT	+	pause in DIBAG session
HELP	+	information about instruction set
KILLBD	+	delete image file
KILLBIN	+	delete mask work-file
KILLHIS	+	delete histogram work-file
KILLLOK	+	delete lookup-table work-file
KILLWORK	+	delete all work files
LOG	+	print work protocol
WORK		information about work-file contents

APPENDIX B

THE IMAGE PROCESSING LANGUAGE

CODES FOR FUNCTIONS IMPLEMENTED IN
THE DVP OPERATIONS PROGRAM DVP

DVPCLR	1	DVPCPY	2	DVPSET	3	DVPCPL	4
DVPSHF	5	DVPSHB	6	DVPXCH	7		
DVPADD	11	DVPSUB	12	DVPMUL	13	DVPDIV	14
DVPMAX	15	DVPMIN	16	DVPCNT	17	DVPSUM	18
DVPA16	19	DVPSQR	20				
DVPADC	21	DVPSBC	22	DVPMLC	23	DVPDVC	24
DVPINC	25	DVPDEC	26	DVPAVG	27	DVPLIN	28
DVPADL	29	DVPTHR	30				
STORCH	31	LOADCH	32	SETRWC	33	SCROLL	34
PIXCHK	35	SETCUR	36	NBMAP	37	NBCNT	38
ZOOM	39	SCRLZM	40				
DVPMAR	41	DVPCLB	42	DVPMCY	43	DVPMCP	44
DVPMND	45	DVPMOR	46	DVPMEO	47	DVPMEQ	48
DVPMSB	49						
DVPABS	51	DVPLOG	52	DVPEXP	53	DVPBOO	54
BOX	61	DVPCMP	62	DVPCME	63		

TO SOFTWARE SWITCHBOARD ... 99

INPUTS REQUIRED

GENERAL PROCEDURE: FIRST PROMPT IS FOR OP-CODE
SECOND PROMPT IS FOR PARAMETERS
^Z IS END

RWC MEANS REGIONAL WRITE CONTROL CODE
0 - UNCONDITIONAL
1 - REGION DEFINED BY CURSORS
2 - REGION DEFINED BY CURSORS AND
RWC BIT OF ITT4
3 - REGION DEFINED BY CURSORS AND
COMPLEMENT OF RWC BIT OF ITT4

```
OP-CODE

DVPCLR ( 1)                    clear image memory
            CHANNEL,RWC

DVPCPY ( 2)                    copy image memory
            SOURCE,TARGET,RWC

DVPSET ( 3)                    set memory to value
            TARGET,VALUE ,RWC

DVPCPL ( 4)                    copy image memory via lookup-table
            SOURCE,TARGET,RWC

DVPSHF ( 5)                    shift 16 bit
            MSB,LSB,SHIFTCOUNT,BYTE/WORD,SHIFT/ROT,OVERFL,RWC

DVPSHB ( 6)                    shift 8 bit
            CHANNEL,SHIFTCOUNT,SHIFT/ROT,OVERFL,RWC

DVPXCH ( 7)                    exchange two images in memory
            CHANNEL1,CHANNEL2,RWC

DVPADD (11)                    addition 8+8 bit
            SUMMAND1,SUMMAND2,SUM,RWC,CLIP

DVPSUB (12)                    subtraction 8-8 bit
            MINUEND,SUBTRAHEND,DIFFERENCE,RWC,CLIP

DVPMUL (13)                    multiply 8 x 8 bit giving 16 bit result
            FACTOR1(BECOMES MSB),FACTOR2,LSB,RWC

DVPDIV (14)                    divide 8 / 8 bit giving 8 bit remainder
            DIVIDEND(BECOMES QUOTIENT),DIVISOR,REMAINDER,RWC

DVPMAX (15)                    maximum of two channels
            OP1,OP2,RESULT,RWC

DVPMIN (16)                    minimum of two channels
            OP1,OP2,RESULT,RWC

DVPCNT (17)                    get count register contents
                OUTPUT: 'COUNTER IS ',F9.0

DVPSUM (18)                    sum over all pixels
            CHANNEL,RWC
                OUTPUT: 'SUM IS ',F9.0

DVPA16 (19)                    addition 16+8 bit
            SUMMAND1,SUMMAND2,OVERFLOW CHANNEL,RWC

DVPSQR (20)                    square root of 16 bit
            MSB,LSB,RESULT,RWC

DVPADC (21)                    addition of 8 bit constant
            SUMMAND,CONSTANT,SUM,RWC,CLIP

DVPSBC (22)                    subtraction of 8 bit constant
            MINUEND,CONSTANT,DIFFERENCE,RWC,CLIP

DVPMLC (23)                    multiply by 8 bit constant
            FACTOR(BECOMES MSB),CONSTANT,LSB,RWC

DVPDVC (24)                    divide by 8 bit constant
```

```
                DIVIDEND(BECOMES QUOTIENT),CONSTANT,REMAINDER,RWC

DVPINC (25)                 increment image memory
                SOURCE,TARGET,RWC

DVPDEC (26)                 decrement image memory
                SOURCE,TARGET,RWC

DVPAVG (27)                 mean of two images
                OP1,OP2,RESULT,RWC

DVPLIN (28)                 scaled linear combination of images
                SOURCE1,FACT1,SOURCE2,FACT2,TARGET,RWC

DVPADL (29)                 addition 8 bit modif by ITT + 8 bit
                CHANNEL1,CHANNEL2,RESULT,RWC

DVPTHR (30)                 threshold on image
                CHANNEL,LOWER LIMIT,UPPER LIMIT,BIT PLANE,RWC
                    OUTPUT: 'AREA IS ',F9.0

STORCH (31)                 save memory to disk file
                SOURCE
                FILE NAME       (FULL NAME WITH EXTENSION REQUIRED)

LOADCH (32)                 load memory from disk file
                TARGET
                FILE NAME       (FULL NAME WITH EXTENSION REQUIRED)

SETRWC (33)                 set regional write control bits in ITT4
                BIT MASK,CLEAR RWCBIT (0/1), SET CURSOR (0/1)

SCROLL (34)                 set scroll for DVP in DIBAG coordinates
                CHANNEL,IDX,IDY

PIXCHK (35)                 check pixel values in memories
                PIXEL IX,IY

SETCUR (36)                 manipulate cursors to fix window
                DIBAG WINDOW    OR <CR> FOR CURSOR INPUT,
                <ENTER> OR <CR> WHEN CURSORS ARE FIXED

NBMAP  (37)                 map neighbourhood in binary mask
                            to 8 bit number
                BIT PLANE,RESULT IMAGE

NBCNT  (38)                 count pixel neighbourhood
                BIT PLANE,RESULT IMAGE,MASK SIZE(ODD,3-15),METRIC

ZOOM   (39)                 zoom
                CHANNEL OR CODE, ZOOM FACTOR (2,4,8)

SCRLZM (40)                 scroll and zoom of specified channels
                CHANNEL OR CODE, ZOOM FACTOR (2,4,8),IX,IY

DVPMAR (41)                 area of mask
                CHANNEL,BIT PLANE,RWC
                    OUTPUT: 'AREA IS ',F9.0

DVPCLB (42)                 clear bit plane
                CHANNEL,BIT PLANE,RWC

DVPMCY (43)                 copy mask (bit plane)
                FROM CHANNEL,BIT PLANE,TO CHANNEL,BIT PLANE,RWC

DVPMCP (44)                 complement mask
```

```
                    SOURCE BIT PLANE,TARGET BIT PLANE

DVPMND (45)                     .and. masks
                    BIT PLANE 1,BIT PLANE 2,TARGET BIT PLANE

DVPMOR (46)                     .or.  masks
                    BIT PLANE 1,BIT PLANE 2,TARGET BIT PLANE

DVPMEO (47)                     exclusive .or. masks
                    BIT PLANE 1,BIT PLANE 2,TARGET BIT PLANE

DVPMEQ (48)                     .equivalence. masks
                    BIT PLANE 1,BIT PLANE 2,TARGET BIT PLANE

DVPMEO (49)                     subtract masks
                    BIT PLANE 1,BIT PLANE 2,TARGET BIT PLANE

DVPABS (51)                     absolute value of 7 bit signed number
                    SOURCE,TARGET,RWC

DVPLOG (52)                     log function on image
                    SOURCE,TARGET,RWC

DVPEXP (53)                     exp function on image
                    SOURCE,TARGET,RWC

DVPBOO (54)                     boolean operations on images
                    CODE,CHANNEL1,CHANNEL2,RESULT,RWC

BOX    (61)                     plot a rectangle (outline or filled)
                    CHANNEL,WINDOW,VALUE,FILL(0/1)

DVPCMP (62)                     set flag mask after comparing
                                two images with 'greater equal'
                    CHANNEL1,CHANNEL2,BIT PLANE,RWC

DVPCME (63)                     set flag mask after comparing
                                two images with 'equal'
                    CHANNEL1,CHANNEL2,BIT PLANE,RWC
```

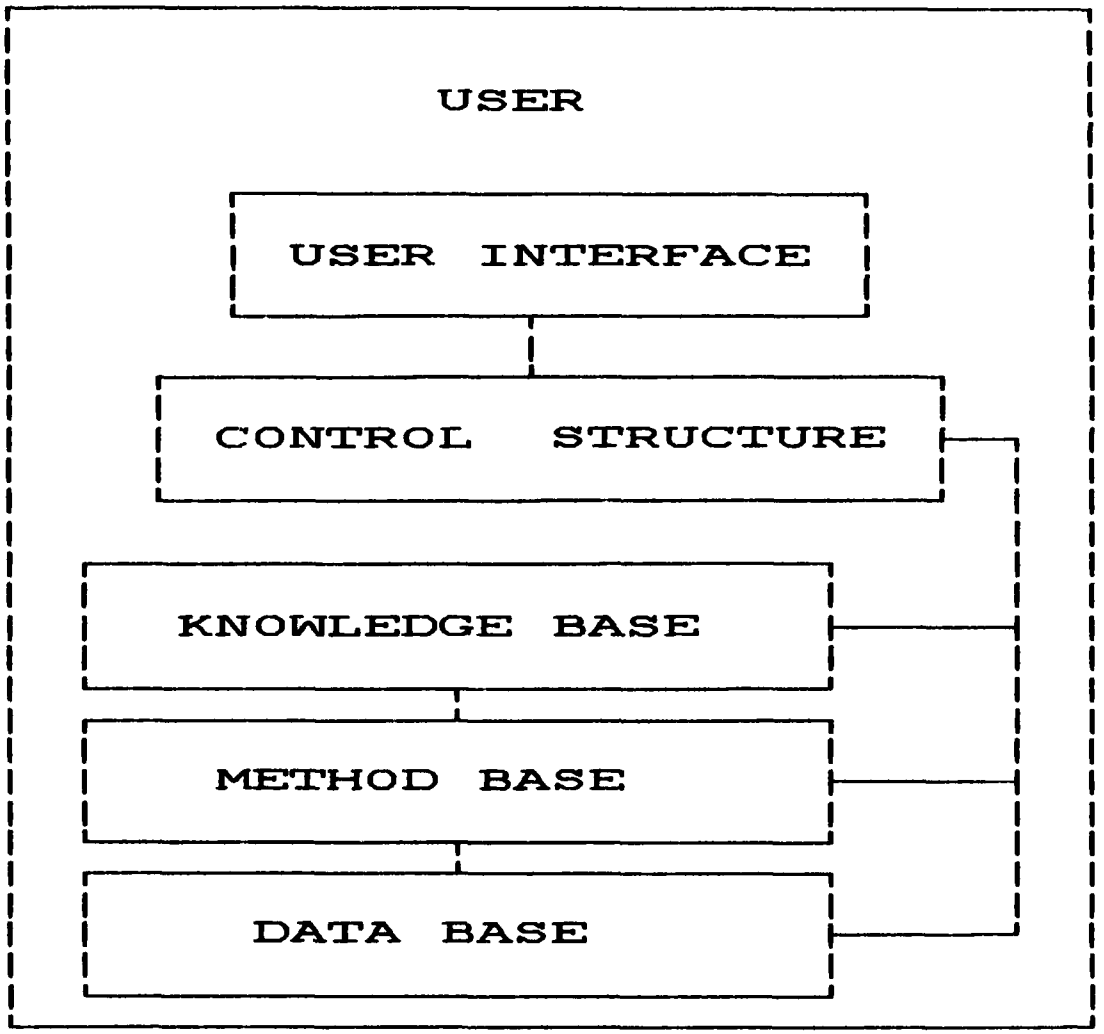


Figure 1: Components of an expert system

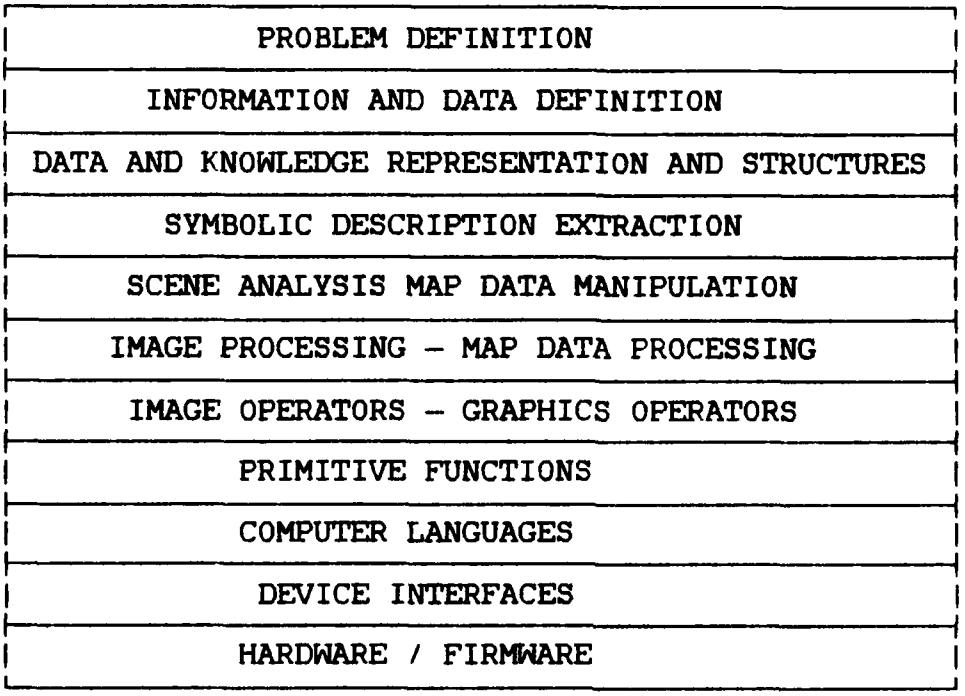


Figure 2: Layers of a computer vision system
for the exploitation of map - image correspondence

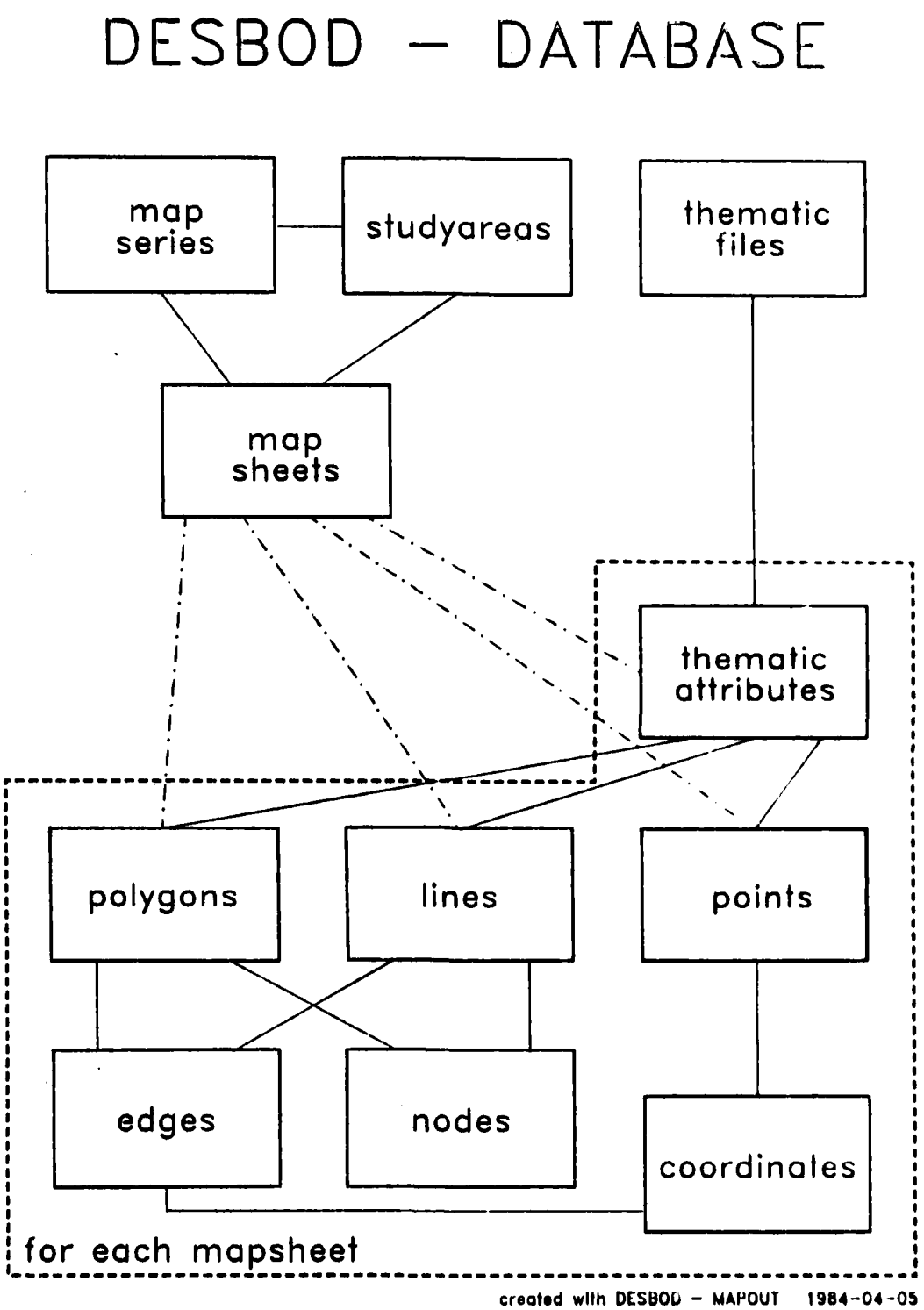


Figure 3: Overview of component elements of the geo-information system DESBOD which is used for map data processing

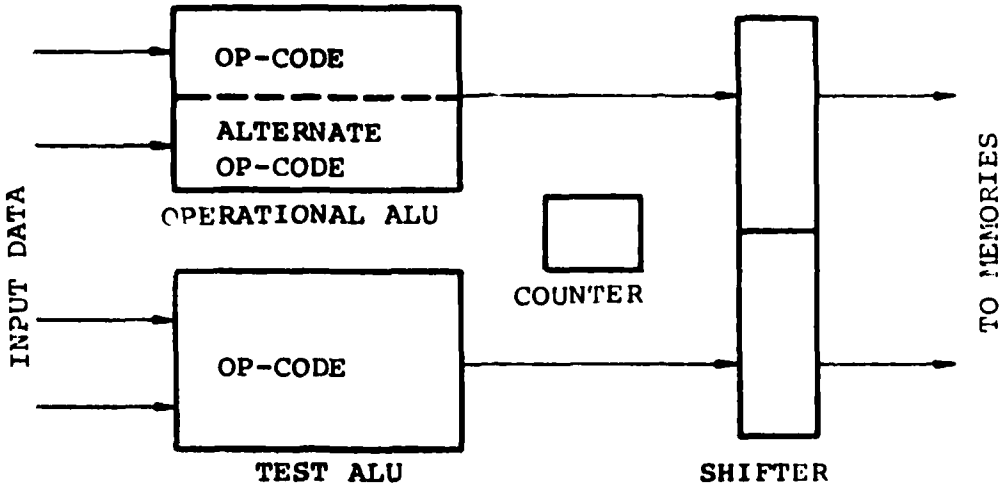


Figure 4: Schematic diagram of the components of the Digital Video Processor DVP

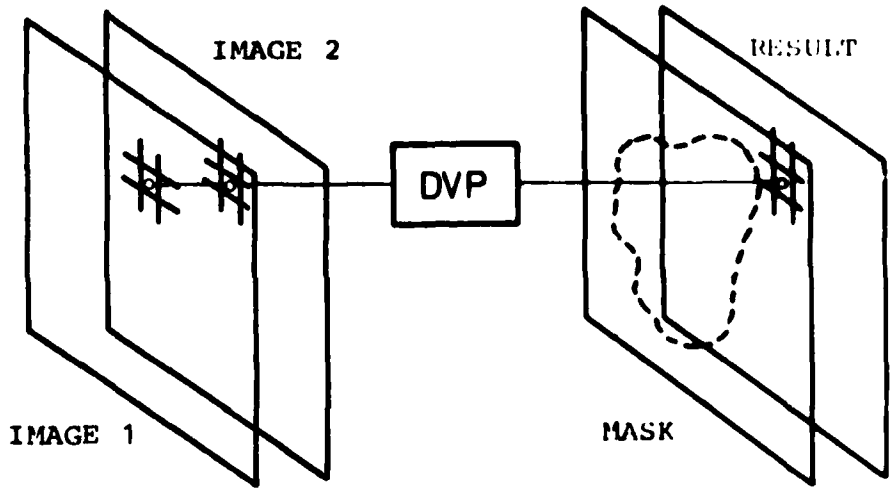
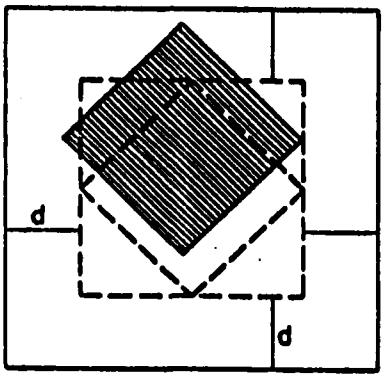


Figure 5: Data flow during an operation cycle of the Digital Video Processor

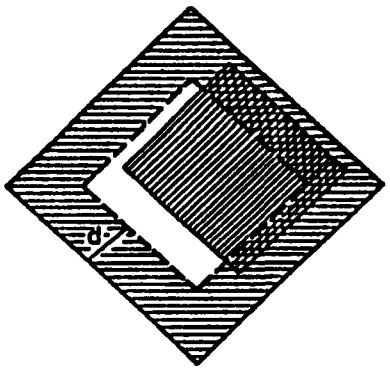


---- projected feature (mask) and its window

//// actual position of the object in the digital image

—— search area

Figure 6: Situation for the recognition algorithms

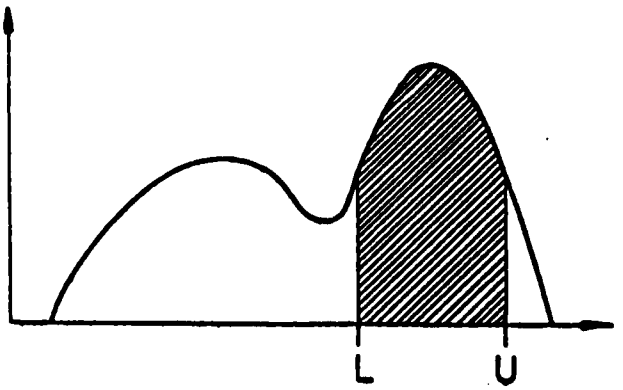


---- projected feature

enlargement of the mask

//// actual position of object

Figure 7: Situation for the algorithm THRESH



//// estimated area of object

L, U lower and upper threshold limits

U - L is minimum for fixed area

Figure 8: Threshold value selection from histogram

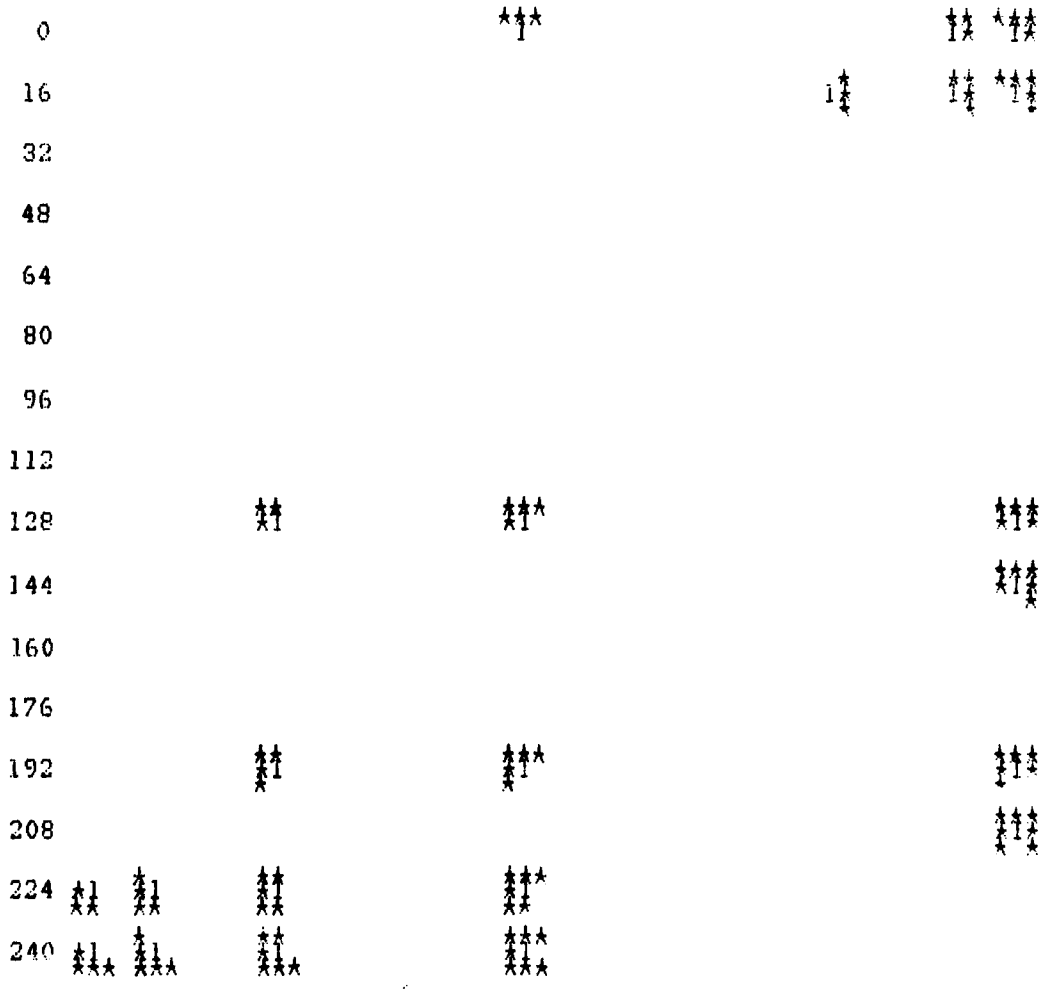


Figure 9 a: Deletable pixels looking south-east

Figure 9 b: Deletable pixels looking north-west

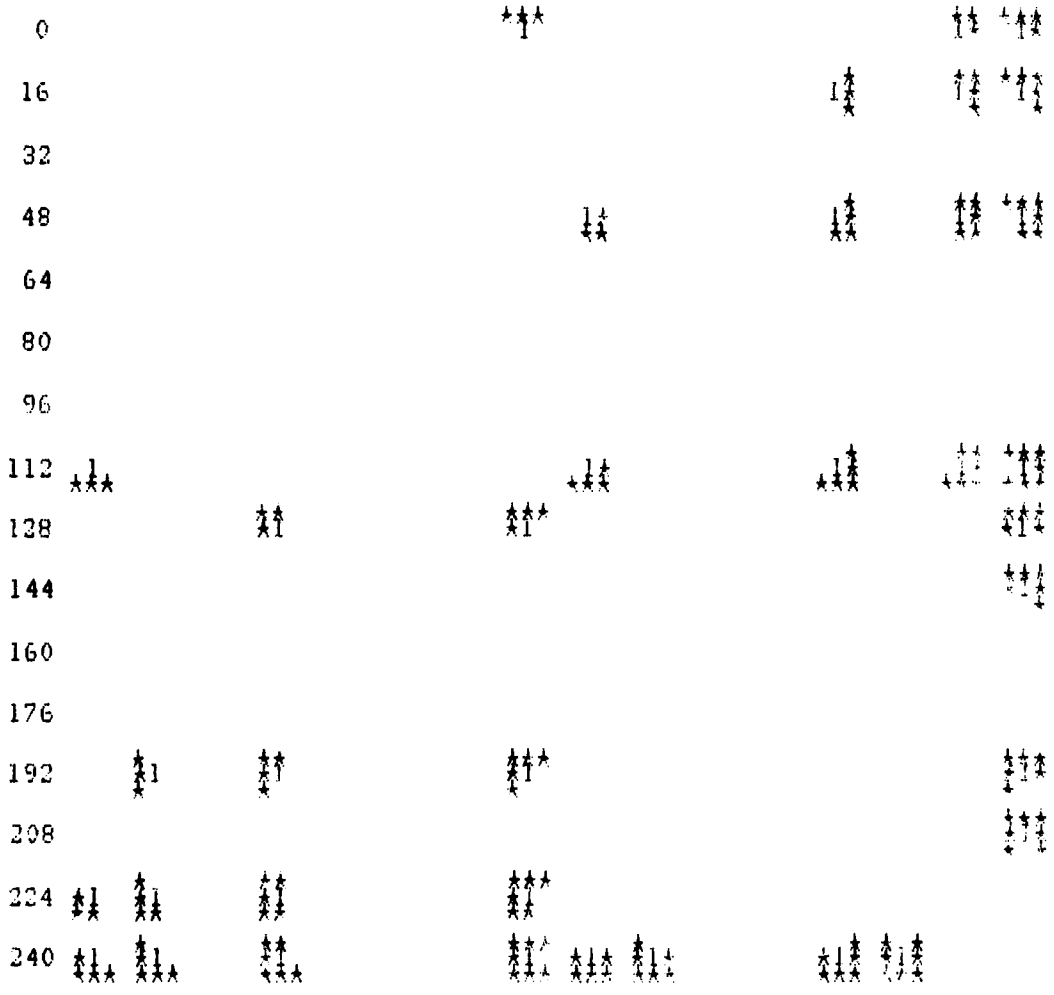


Figure 9 c: Deletable simple pixels in 4-neighbourhood

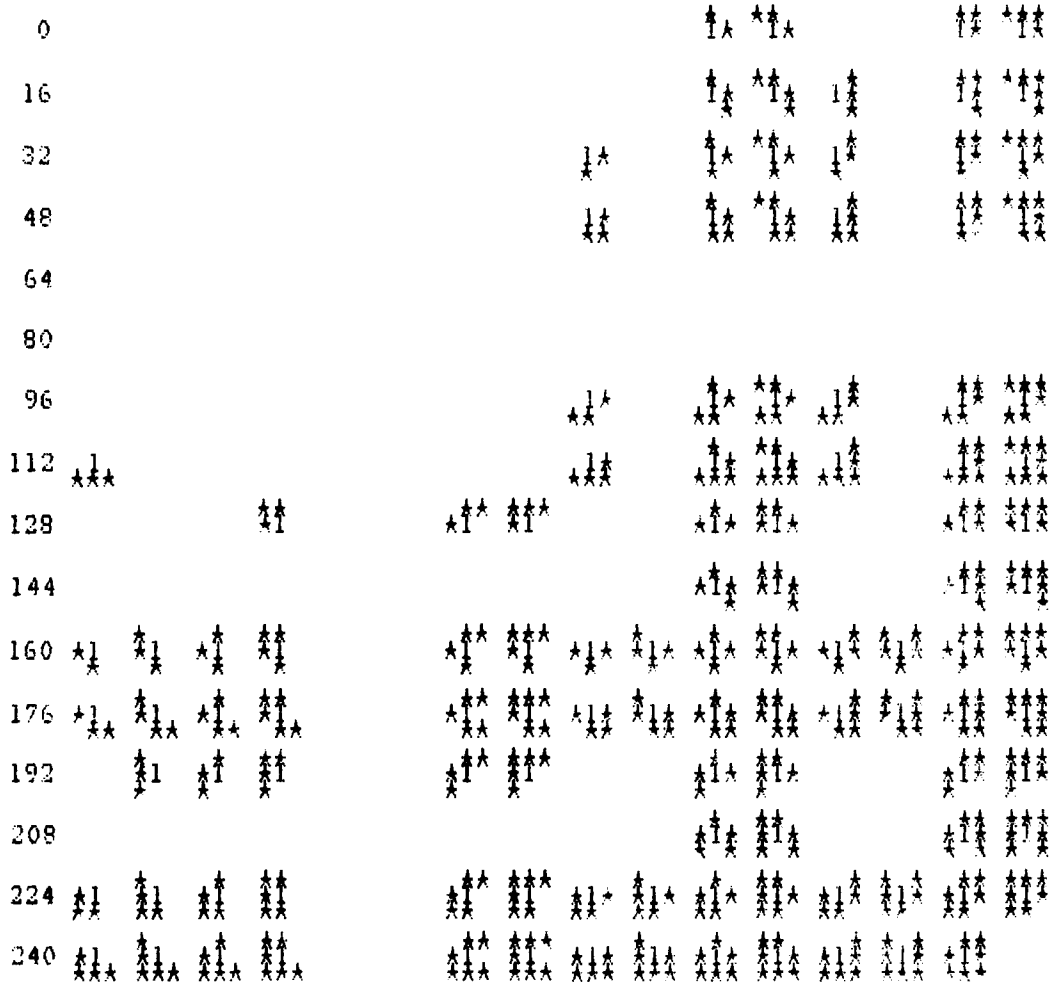
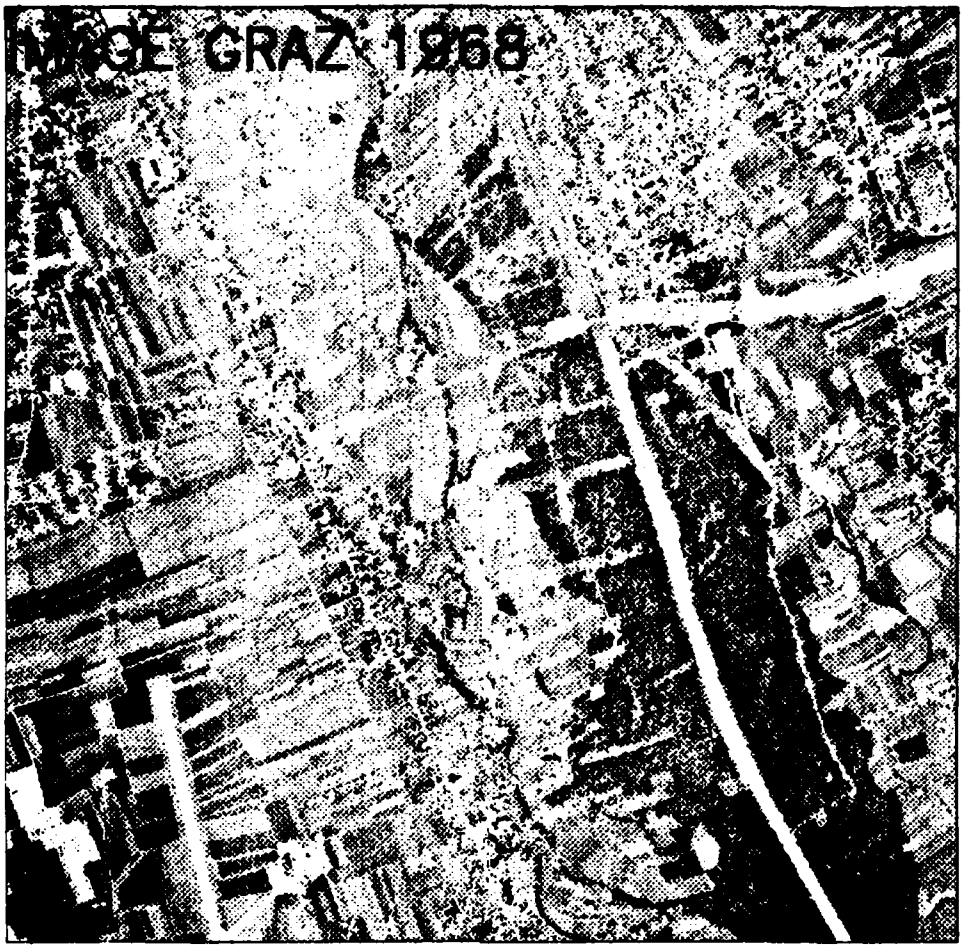


Figure 9 d: Deletable simple pixels in 8-neighbourhood



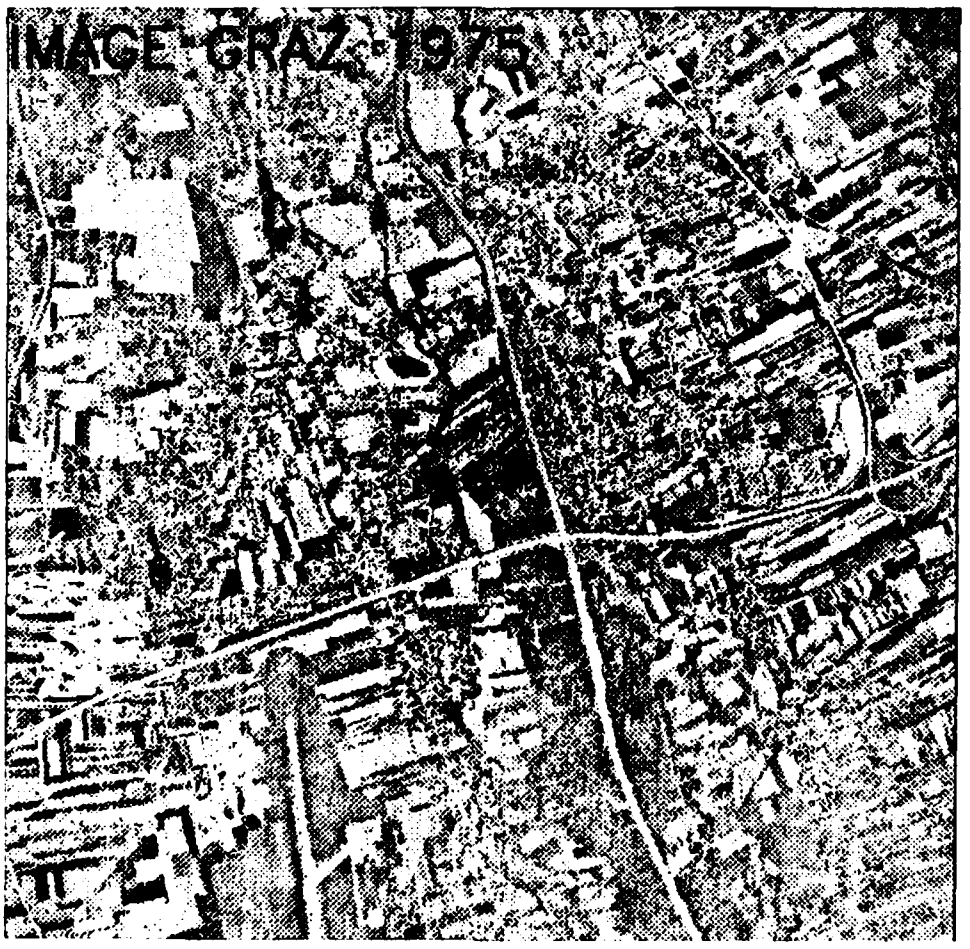
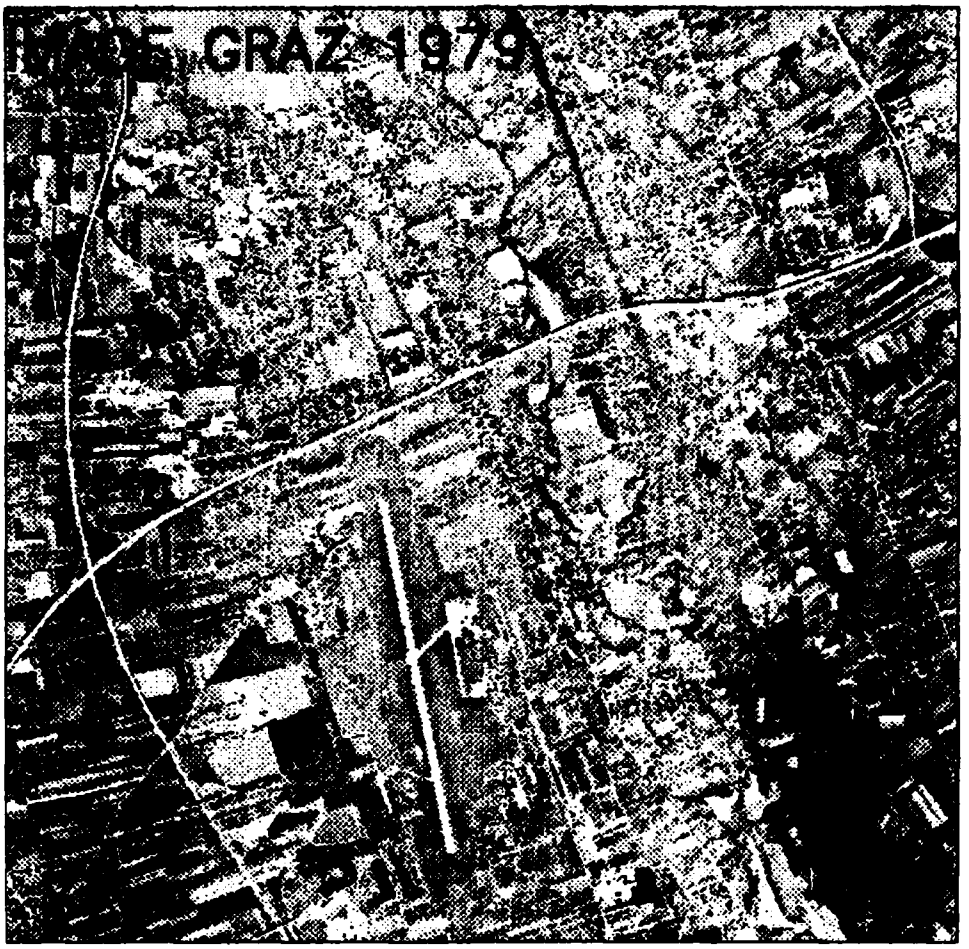


Figure 10: The multitemporal set of digitized aerial photographs



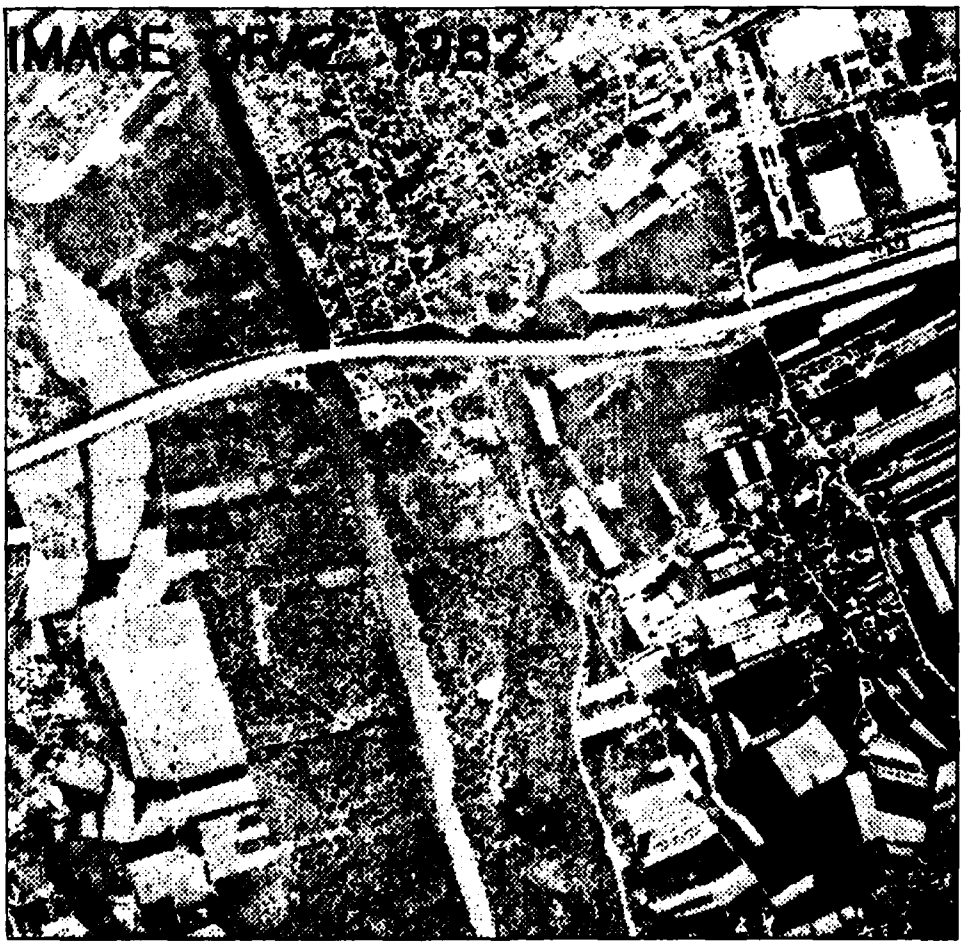


Figure 10 continued

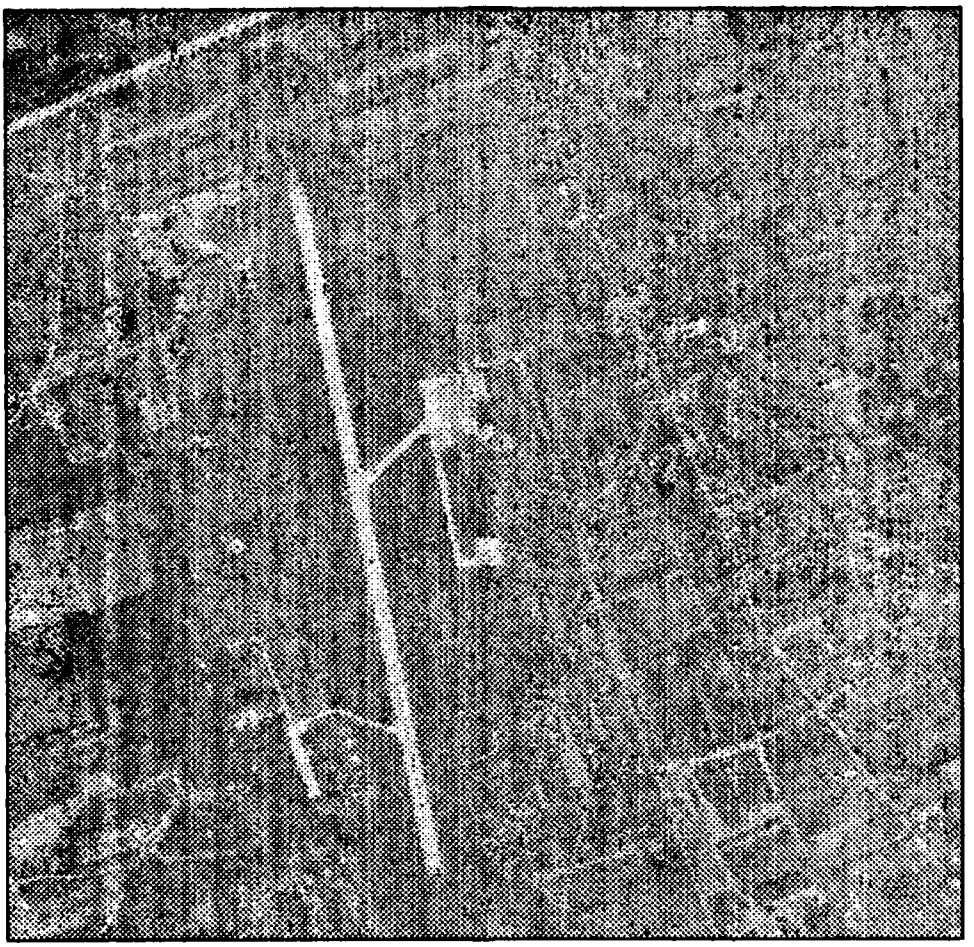


Figure 11: An airport scene, with the original scanner values mapped to the gray scale of the electrostatic plotter. The appearance on the video monitor of the image processing device is good, but transformations have to be applied to also yield an acceptable output on the plotter.

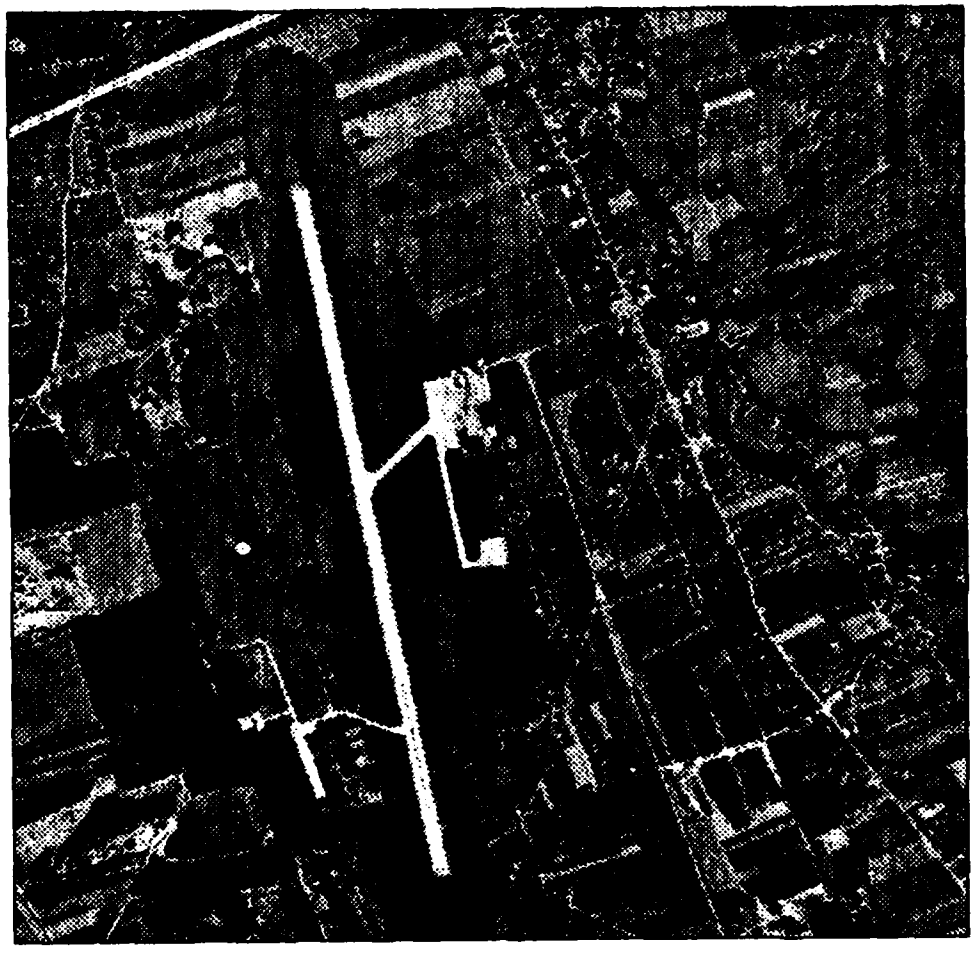


Figure 12: The airport scene of figure 11, enhanced for optimal output on the electrostatic plotter. First a linear stretch is performed, then the image is requantized to three bit resolution (DEANZA programs ENHANC and REDRES employing the DVP for computation of an optimal ramp). The example shows how well the instruction set fits the need for various gray value transformations necessary to optimize imagery with respect to various output devices.

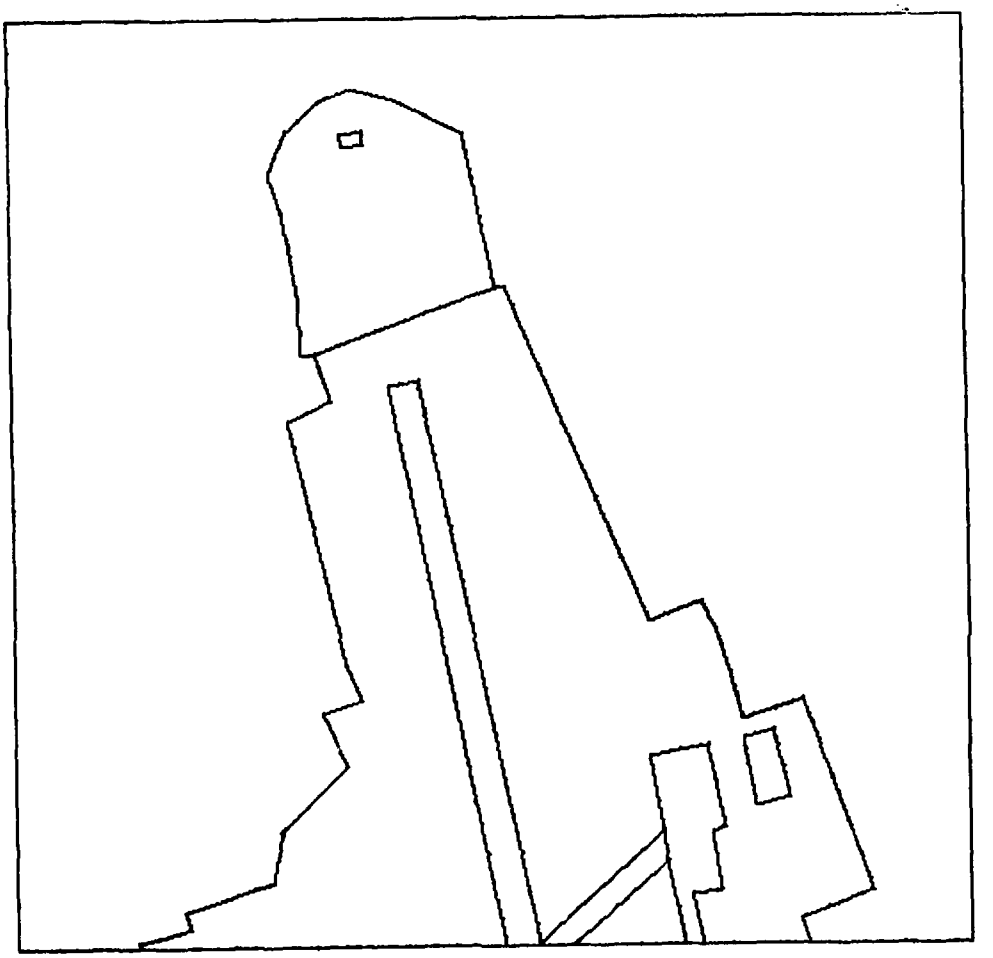


Figure 13: Part of a map depicting features
at the airport

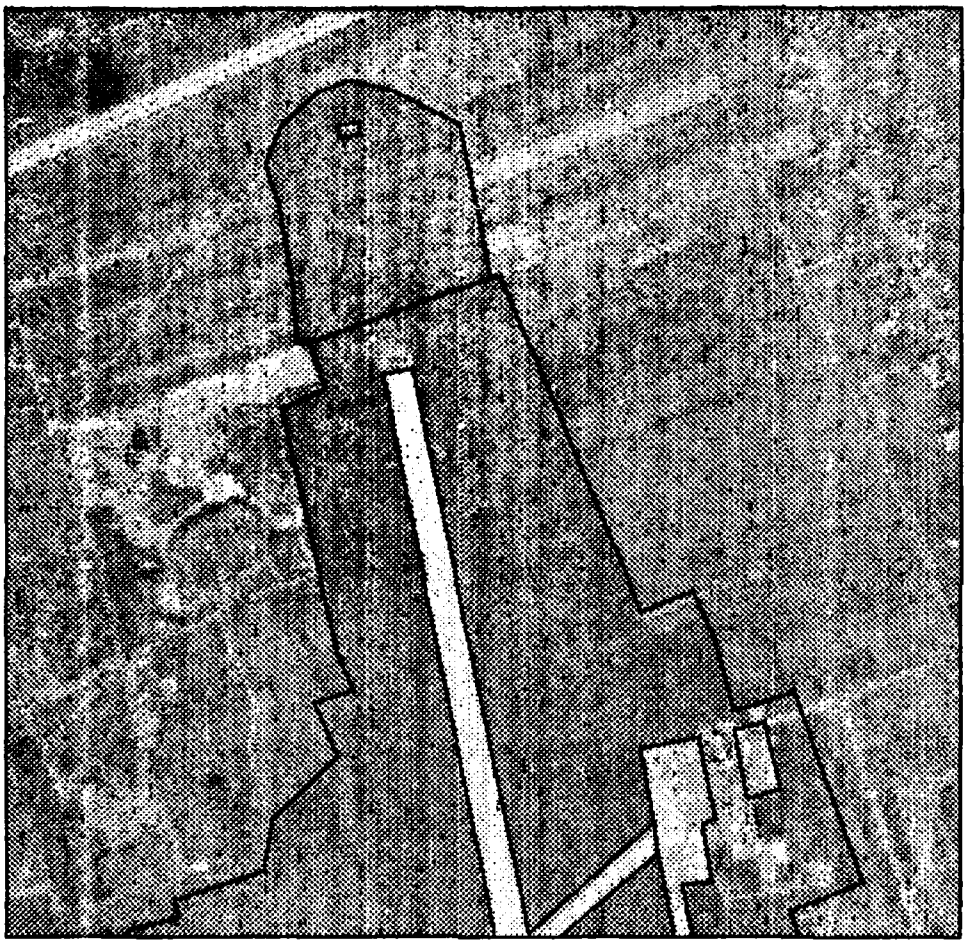


Figure 14: Part of the airport scene with the map superimposed by applying a warping function based on control points.

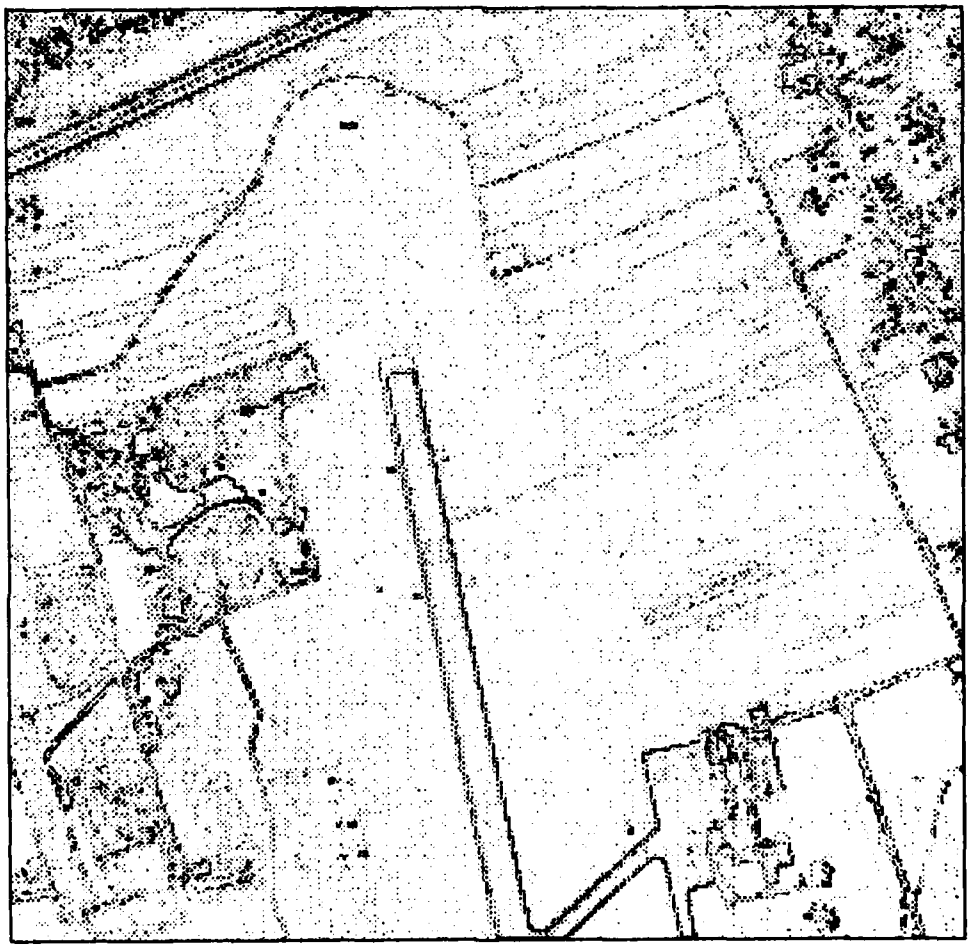


Figure 15: Roberts edge image (gradient magnitude) computed on the DVP as an approach to establish image-map correspondence. Compare with figure 13.

CONTROL PLOT OF FILE CZSMAP
CREATED ON 4-DEC-84 AT 18:36

IBM
INSTITUTE FOR IMAGE PROCESSING
AND COMPUTER GRAPHICS

Figure 16: Linear structures digitized and stored in DESBOD in a general coordinate system. The correspondence to the image coordinate system is established via control points.

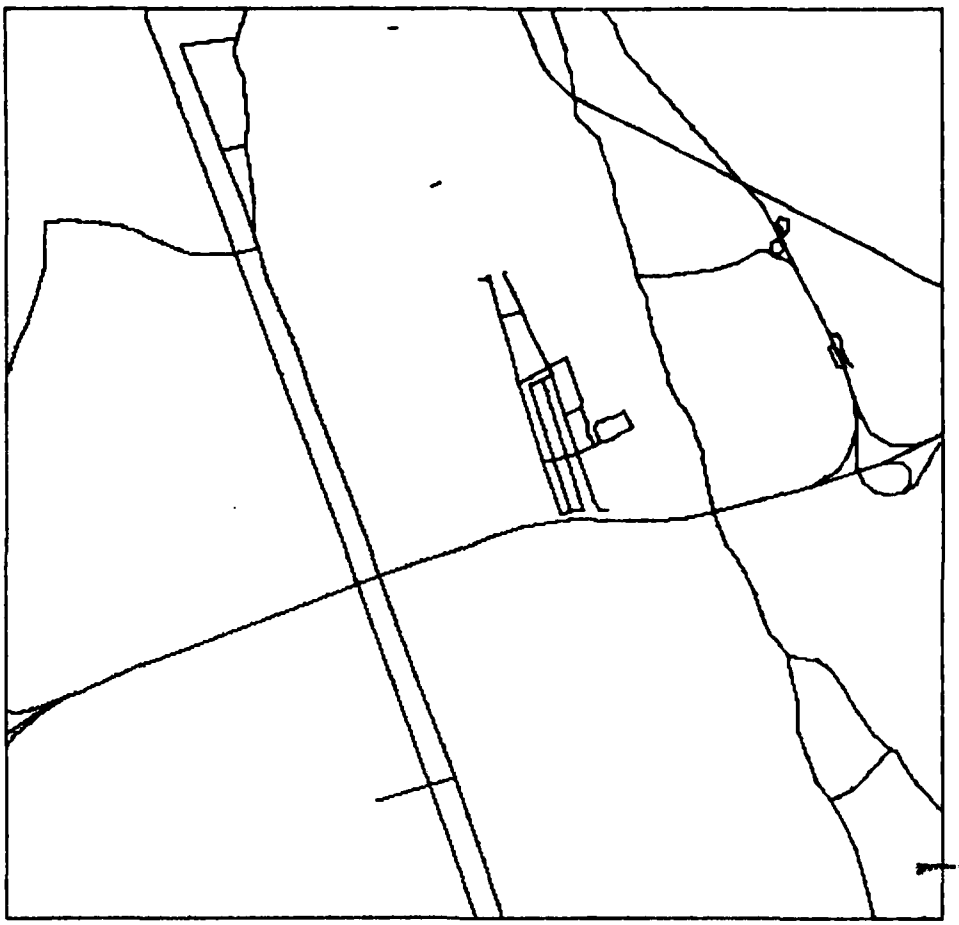


Figure 17: The linear structures of figure 16 after transformation and vector-to-raster conversion.

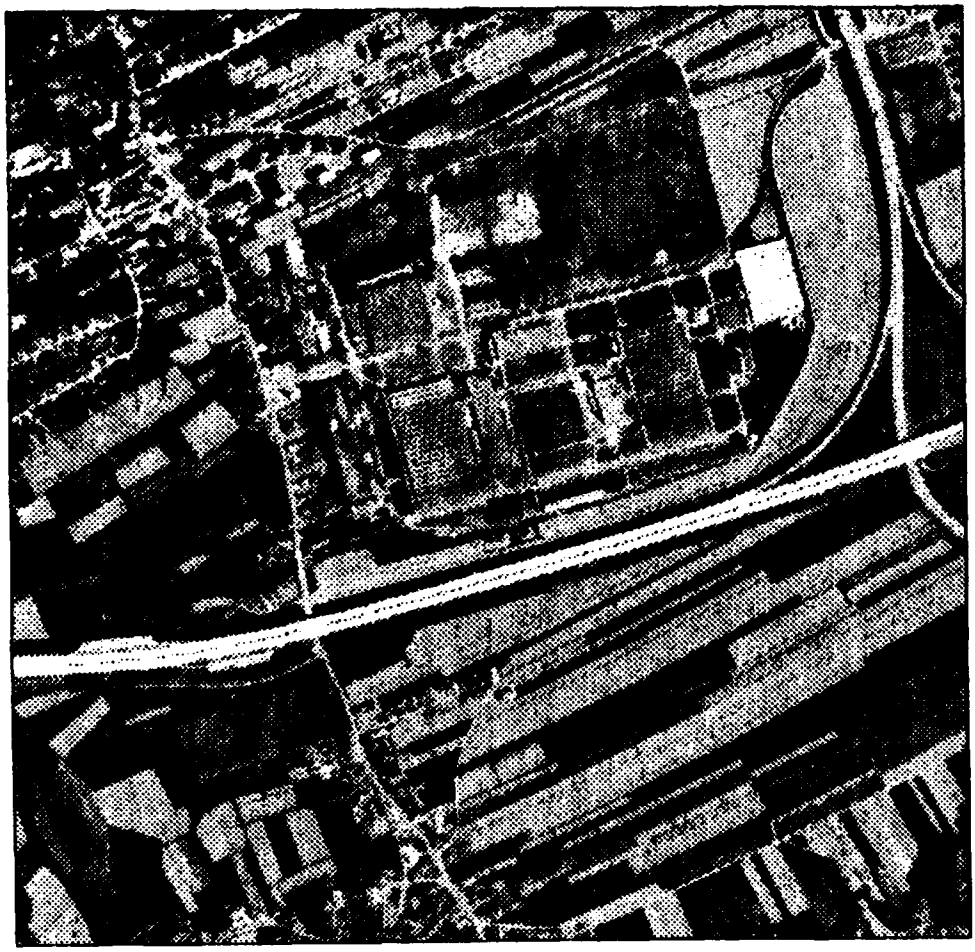


Figure 18: Subscene from GRAZ75 depicting a factory and agricultural areas (window 1481, 901, 512, 512).

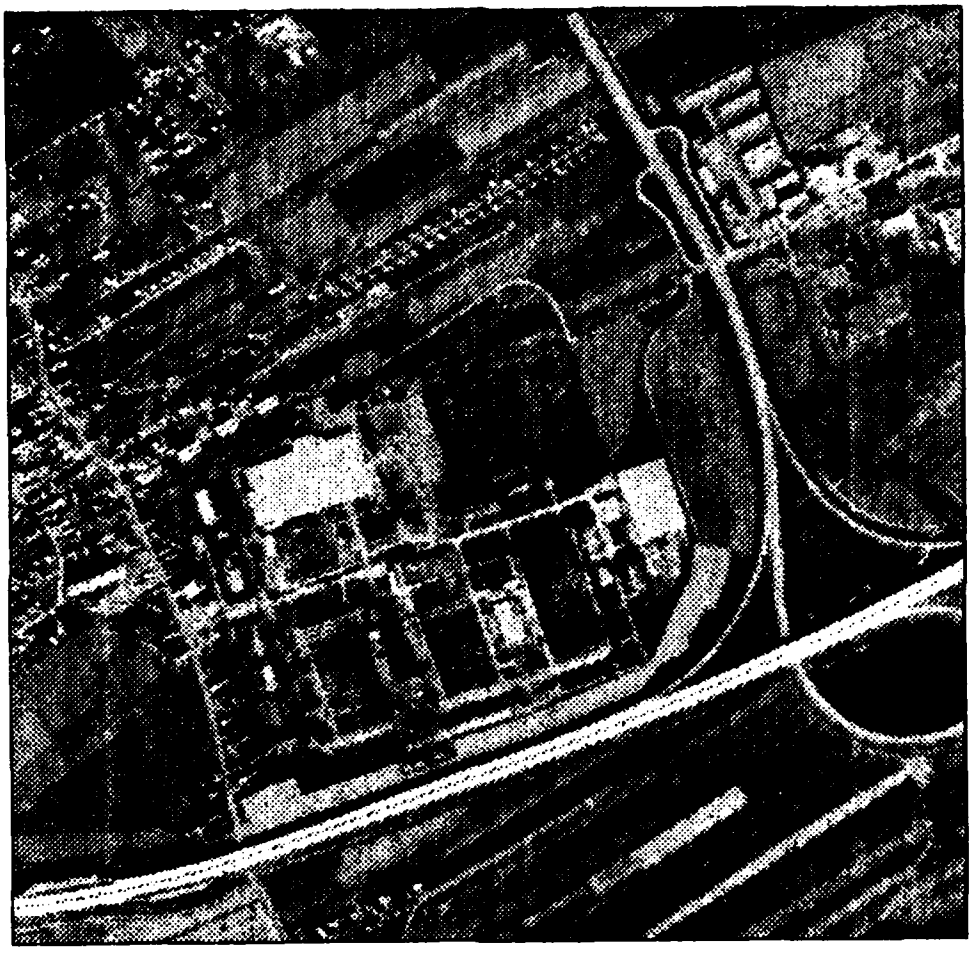


Figure 19: The approximately corresponding subscene from GRAZ79 (window 1637, 256, 512, 512).

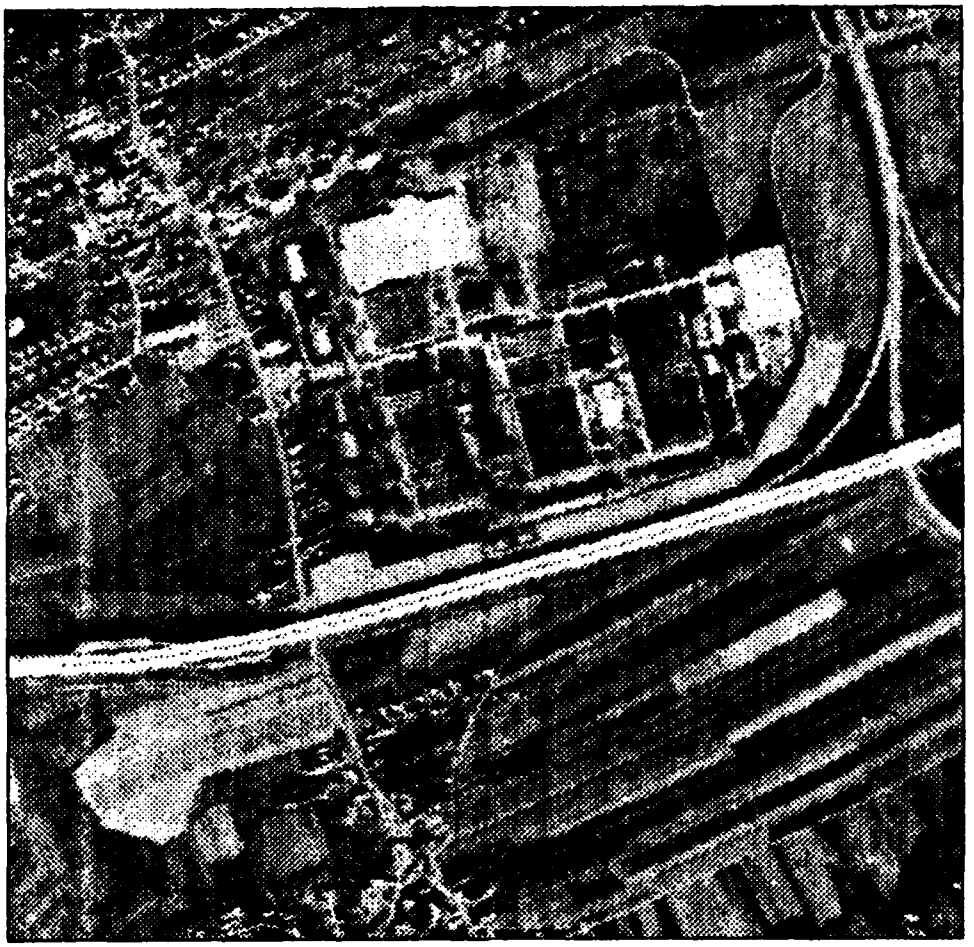


Figure 20: Subscene from GRAZ79 rectified to the geometry of GRAZ75. Note the various agricultural patterns and their changes between 1975 and 1979 by comparison with figure 18.

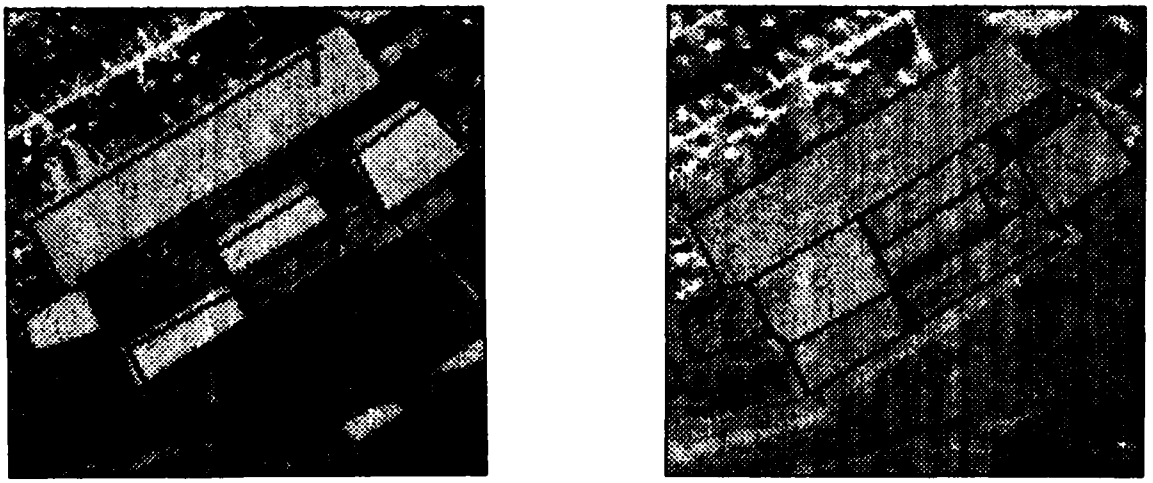


Figure 21: (a) An agricultural pattern from GRAZ75 with the superimposed map; (b) The same pattern from GRAZ79. Note that many areas are merged now.
(window 1,191,128,128)

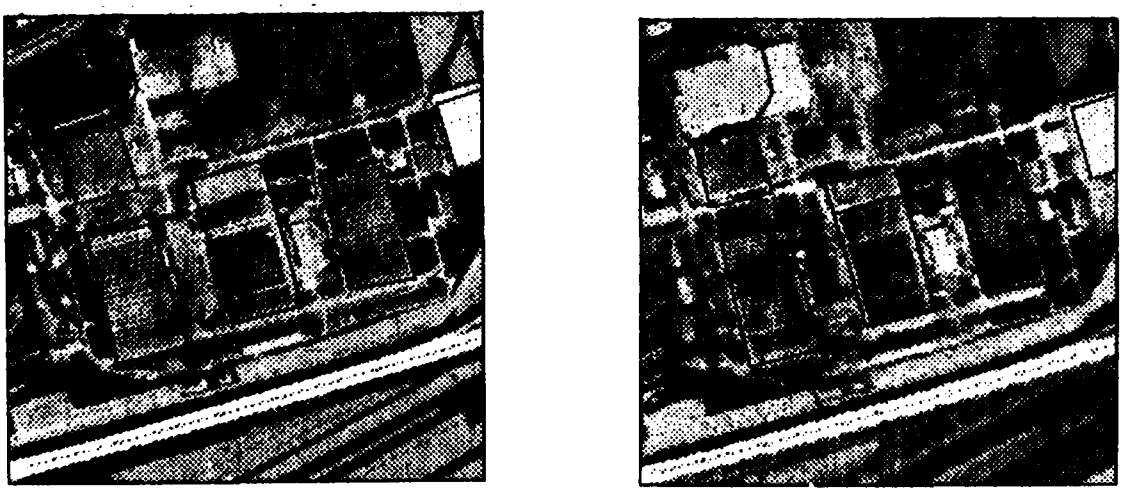


Figure 22: (a) Factory from GRAZ75 with the superimposed map; (b) Factory from GRAZ79 which was rectified to the geometry of GRAZ75. In the area at the upper left corner, a new building was erected at the site of a former park. This area and the rightmost building were used for further experiments.
(window 161,91,256,256)

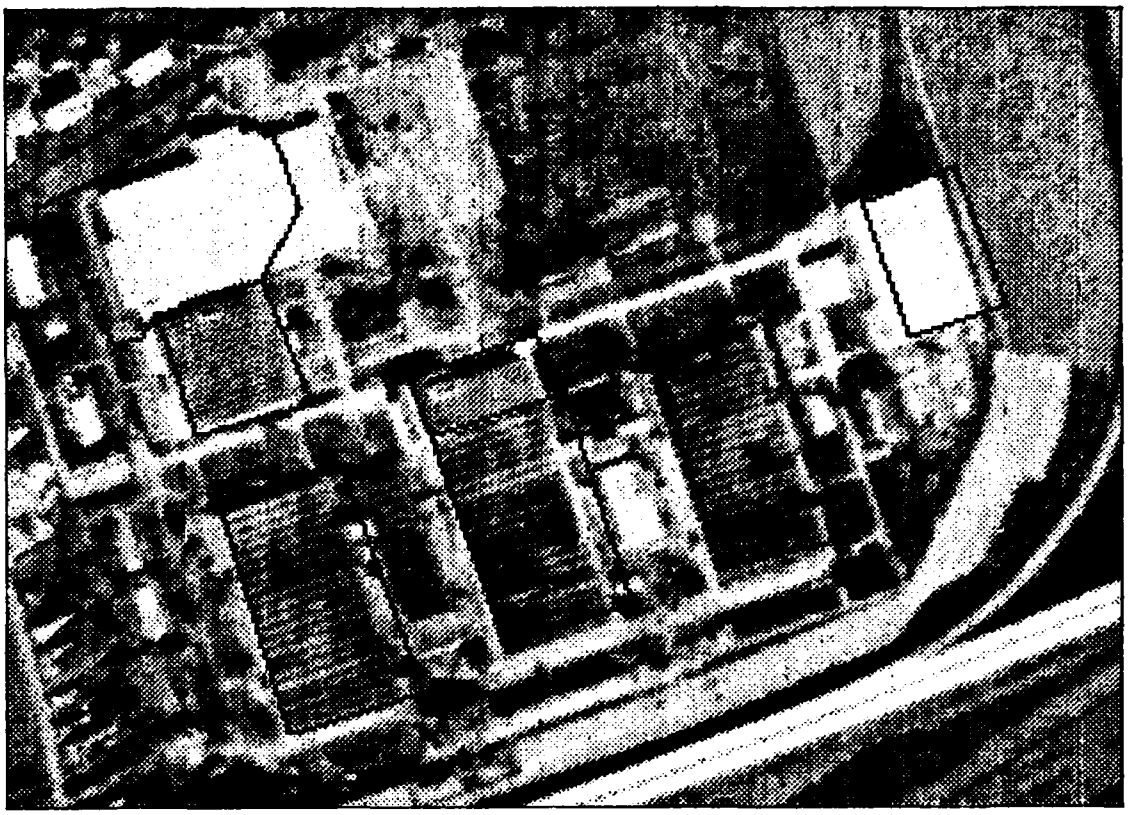


Figure 23: Factory area as seen on the original GRAZ79 image with the superimposed map. This is the typical situation when initiating the recognition algorithms. (window 101,201,300,200)

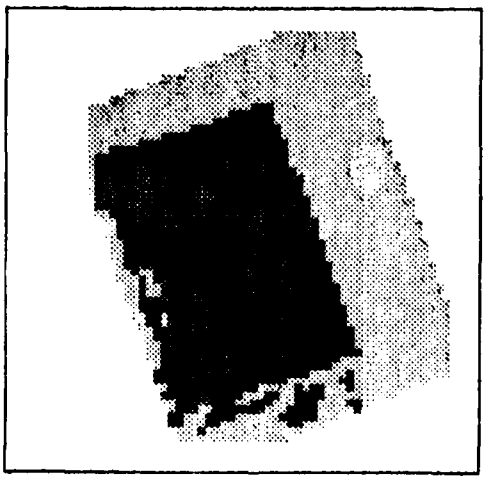


Figure 24: Example depicting the approach of algorithm THRESH (see also figure7): The enlarged mask is shown in light gray, the result of thresholding in dark gray. Subsequent smoothing leads to the final result shown in figure 25(a).

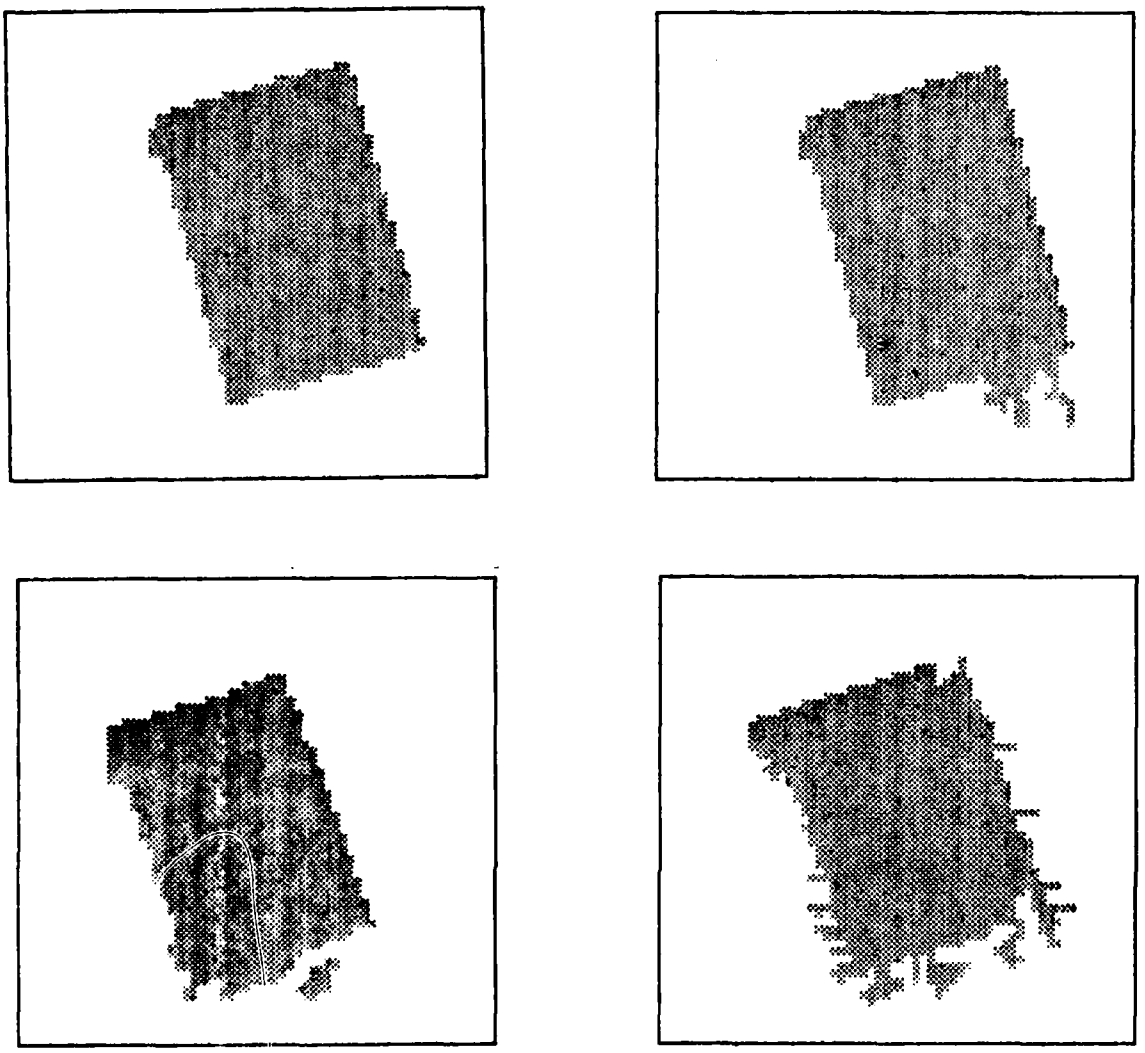


Figure 25: The factory building recognized by the algorithms at 1975 and 1979. (a) THRESH, GRAZ75, smoothed with region growing. (b) ADAPT, GRAZ75; lacking contrast at the south results in the inclusion of non-object pixels. SHIFT gives a displacement vector of (-2,-4) and a similarity measure of 0.4494. (c) THRESH, GRAZ75, smoothed by mask transformation; a larger exterior area remains as noise. (d) ADAPT, GRAZ79; the sensitivity of the algorithm with respect to small gray value changes in the image becomes apparent in the rugged border. SHIFT resulted in a vector of (-5,+4) and c=0.2387. (window for a and b was 371,121,64,64; for c and d, 315,236,64,64)

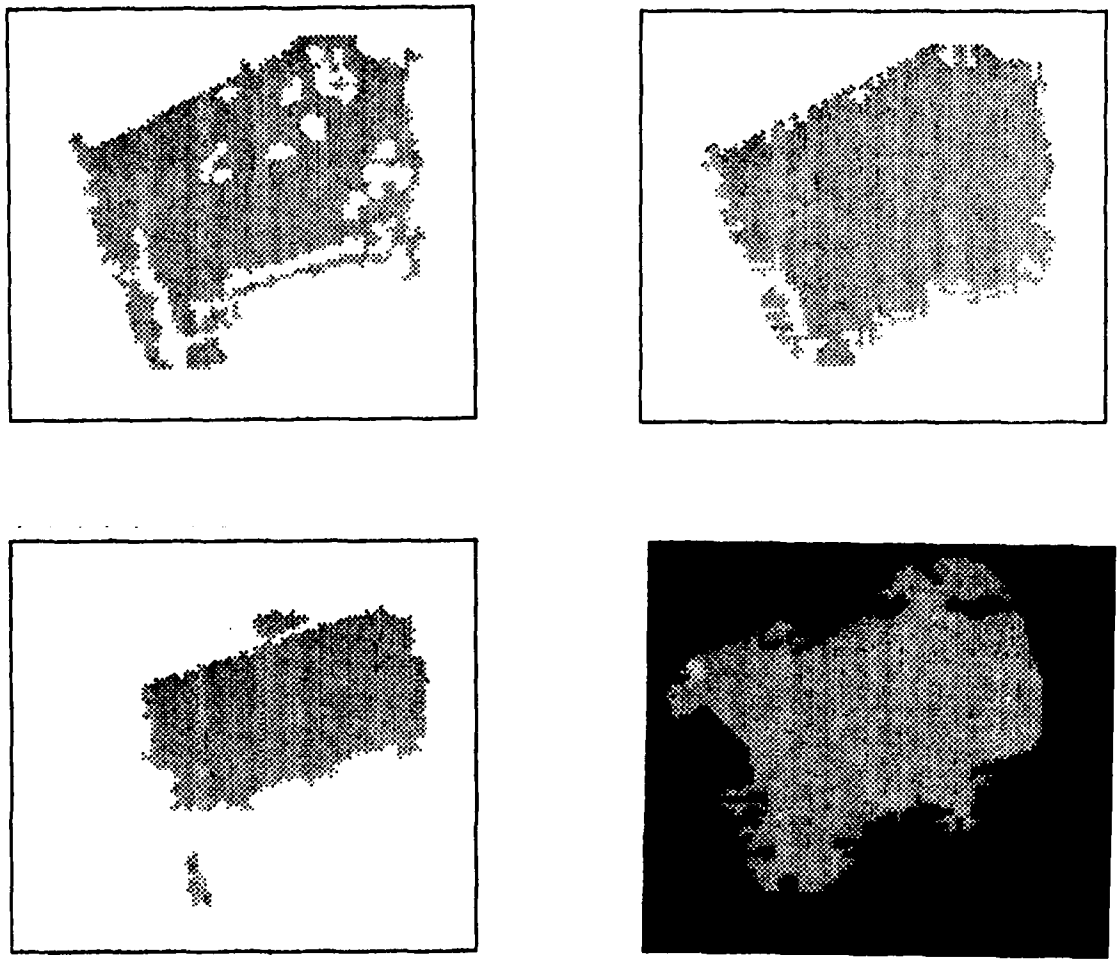


Figure 26: The park area from 1975 which in 1979 had become a building. (a) THRESH, GRAZ75; larger interior areas not indicated on the map (mostly bare soil in the park) were not closed in the smoothing process. (b) ADAPT, GRAZ79; the rugged border is visible again. SHIFT gave a vector of (-2,-3) and similarity was 0.1797. (c) THRESH, GRAZ79; the object detected, namely the new building, clearly is dissimilar to what was expected and the area is flagged as having changed. (d) ADAPT, GRAZ79; this algorithm shows a tendency to keep the old form and verification of change fails. SHIFT gave a displacement of (+5,+5) and an exceptionally low value of c=0.1319. (windows shown are 141,91,100,90 and 101,221,100,90)

END

FILMED

9-85

DTIC